Ecom-Exchange Communication Point Modelling in the Context of Enterprise Architecture

Patricia Cecilia Lopes Semedo

Thesis to obtain the Master of Science Degree in Telecommunications and Informatics Engineering

Supervisor: Prof. André Ferreira Ferrão Couto e Vasconcelos

Examination Committee

Chairperson: Prof. Ricardo Jorge Fernandes Chaves
Supervisor: Prof. André Ferreira Ferrão Couto e Vasconcelos
Member of the Committee: Prof. Fernando Henrique Córte-Real Mira da Silva

June 2019
Tudo o que tu podes alcançar está à distancia da tua coragem para arriscar [1]
Acknowledgments

I would like to thank all the individual, work and support that helped me directly or indirectly to complete this work. Firstly, I would like to acknowledge my dissertation supervisor André Vasconcelos who played a very important role in making this thesis possible with his expertise guidance, valuable suggestions, constructive critiques and continued support throughout this journey. I would also like to thank my friends and family for constantly encouraging and motivating me not to give up on my dreams and to keep working harder to achieve it. Secondly, I would also like to extend my sincere appreciations and gratitude to my mother and grandmother for always supporting and making this dream come true, without their help this achievement will never be possible. I am also highly indebted to all my colleagues that helped me grow as person during all these years in the university.

To each and every one of you – Thank you.
Internet eXchange Point (IXP) provides an infrastructure which allows a set of organisations to share information according to the service level agreement establish by them. Since these systems are complex, it is very important to understand its performance and operation in order to improve the management and to reduce the cost associated with implementation and the information shared. The constant evolution of these systems has made it possible for the organisation to have more coherent concepts and representations. Enterprise Architecture (EA) has a set of methods and models for the design of system, modulation of the processes regarding business domain, network infrastructure and the different application running. The EA modelling language have only the general concept of network, if you want to specify the information more specific such as protocols used for share information, the internet protocols or the network address used for sharing information you have to adapt the concepts already existing or create new ones. The nature of this dissertation is the addition of new concepts to the technology layer of reference language (ArchiMate) to enhance the representation and management of the network infrastructure. The new concepts that is going to be specialised are router, switch, interface router, interface switch and routing table. with the addition of these elements it will be possible to represent the Internet Protocol (IP) address, Media Access Control (MAC) address, protocol for exchanging the traffic and the information about the different organisations connected to the IXP. For the evaluation of these concepts, a Portuguese IXP was implemented in the ArchiMate. A case was developed for testing if something fails in the network what process can be affect.

Keywords

Internet Exchange Point, Enterprise architecture, ArchiMate, Concepts, Elements, Meta-Modelling, Modelling Extension.
Resumo

Um Ponto de Troca de Tráfego (PTT) disponibiliza uma infraestrutura que permite várias organizações partilharem informações de acordo com as regras de roteamento e políticas estabelecidas entre eles. Estes sistemas envolvem muita complexidade, na medida que temos de perceber como fazer a gestão dos vários conceitos associados de forma a reduzir os custos associados, por uma melhor implementação e partilha. A evolução constante da tecnologia faz com que as organizações sintam a necessidade de representar esses conceitos de uma forma estruturada. Existem arquiteturas empresariais que usam um conjunto de métodos e modelos para o desenho dos sistemas, implementação dos sistemas e da estrutura associado a uma tecnologia de informação. Essas arquiteturas empresariais não são flexíveis o suficiente de forma a acompanhar a complexidade dos sistemas de redes que estão constantemente a evoluir. Existe a necessidade de haver arquiteturas empresariais que possam representar mais rigorosamente a informação sobre uma rede tecnológica. A investigação proposta, tem como objetivo a especialização de conceitos da camada tecnológica da linguagem de modelação ArchiMate. Os novos elementos introduzidos são router, interface router, interface switch, routing table. Com a introdução desses novos elementos será possível representar o IP address, MAC address, as diferentes interfaces do router, as diferentes interfaces do switch e os endereços dos diferentes caminhos que são usados para enviar os pacotes para os diferentes destinos. Para a avaliação dos novos elementos introduzidos será implementada o caso de Ponto de Troca de Trafego que conecta os serviços publicos em Portugal. Depois será feita um caso de uso para avaliar quais são as vantagens de implementar um PTT numa arquitetura empresarial.

Palavras Chave

Ponto de Troca de Tráfego, Arquitetura Empresarial, ArchiMate, conceitos, elementos, Modulação.
# Contents

1 Introduction .................................................. 1
   1.1 Context .................................................. 3
   1.2 Objectives ................................................. 4
   1.3 Thesis Contributions ...................................... 4
   1.4 Definition of the Problem ................................ 5
   1.5 Research Methodology .................................... 5
   1.6 Outline of the Document .................................. 6

2 Related Work .................................................. 7
   2.1 Internet Exchange Point .................................. 9
   2.2 Exchange Points In The World .......................... 10
   2.3 Network Planning and Dimensioning ..................... 12
      2.3.1 Border Gateway Protocol ......................... 12
      2.3.2 Network Elements ................................. 13
   2.4 Enterprise Architecture ................................. 14
      2.4.1 ArchiMate ......................................... 15
      2.4.1.A Business Layer .................................. 16
      2.4.1.B Application Layer .............................. 16
      2.4.1.C Technology Layer .............................. 17
      2.4.1.D ArchiMate Meta-Model ......................... 18
      2.4.2 System Modelling Language ....................... 21
      2.4.3 Yet Another Next Generation ..................... 22
      2.4.4 Network Description Language ................... 23
      2.4.5 Directory Enabled Network next generation .... 26
      2.4.6 Comparative Analysis ............................. 26
   2.5 Enterprise Modelling Languages ....................... 29
      2.5.1 ArchiMate ......................................... 15
      2.5.1.A Business Layer .................................. 16
      2.5.1.B Application Layer .............................. 16
      2.5.1.C Technology Layer .............................. 17
      2.5.1.D ArchiMate Meta-Model ......................... 18
      2.5.2 System Modelling Language ....................... 21
      2.5.3 Yet Another Next Generation ..................... 22
      2.5.4 Network Description Language ................... 23
      2.5.5 Directory Enabled Network next generation .... 26
      2.5.6 Comparative Analysis ............................. 26
   2.6 Language Extension Mechanism ......................... 27
   2.7 Proposed Extensions In Enterprise Architecture ....... 29
### List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Phases of Design Science Research</td>
<td>6</td>
</tr>
<tr>
<td>2.1</td>
<td>Architecture of Internet Exchange Poin [2]</td>
<td>10</td>
</tr>
<tr>
<td>2.2</td>
<td>(Packet clearing House) Exchange traffic point in the world</td>
<td>11</td>
</tr>
<tr>
<td>2.3</td>
<td>Example of a external BGP (eBGP) and internal BGP (iBGP) between router Autonomous Systems (AS) [3]</td>
<td>13</td>
</tr>
<tr>
<td>2.4</td>
<td>Layers of an Enterprise Architecture</td>
<td>14</td>
</tr>
<tr>
<td>2.5</td>
<td>Interconnection of different concepts of ArchiMate</td>
<td>15</td>
</tr>
<tr>
<td>2.6</td>
<td>Example of a network implemented in ArchiMate</td>
<td>18</td>
</tr>
<tr>
<td>2.7</td>
<td>Example of ArchiMate Meta-model</td>
<td>19</td>
</tr>
<tr>
<td>2.8</td>
<td>Example of Technology Layer meta-model</td>
<td>19</td>
</tr>
<tr>
<td>2.9</td>
<td>Legends of arrows in ArchiMate</td>
<td>20</td>
</tr>
<tr>
<td>2.10</td>
<td>Pillars of System Modeling Language (SysML)</td>
<td>22</td>
</tr>
<tr>
<td>2.11</td>
<td>Resume of the Network Description language (NDL) [4]</td>
<td>25</td>
</tr>
<tr>
<td>2.12</td>
<td>Example of a network implementing in NDL</td>
<td>25</td>
</tr>
<tr>
<td>2.13</td>
<td>Example of a telecom extension for ArchiMate technology layer meta-model [5]</td>
<td>30</td>
</tr>
<tr>
<td>2.14</td>
<td>Architecture of MEGa-modeling Architecture Frameworks (Megaf)</td>
<td>31</td>
</tr>
<tr>
<td>2.15</td>
<td>Meta-model of ARMOR</td>
<td>31</td>
</tr>
<tr>
<td>3.1</td>
<td>New proposed ArchiMate Technology model</td>
<td>36</td>
</tr>
<tr>
<td>3.2</td>
<td>Router Notation</td>
<td>37</td>
</tr>
<tr>
<td>3.3</td>
<td>Switch Notation</td>
<td>37</td>
</tr>
<tr>
<td>3.4</td>
<td>Routing Table Notation</td>
<td>38</td>
</tr>
<tr>
<td>3.5</td>
<td>Interface Switch Notation</td>
<td>38</td>
</tr>
<tr>
<td>3.6</td>
<td>Interface Router Notation</td>
<td>39</td>
</tr>
<tr>
<td>3.7</td>
<td>New Proposed ArchiMate Meta-Model</td>
<td>41</td>
</tr>
<tr>
<td>4.1</td>
<td>Example of git proprieties</td>
<td>46</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>4.2 Example of Clone URI.</td>
<td>46</td>
<td></td>
</tr>
<tr>
<td>4.3 Example of repository URI.</td>
<td>47</td>
<td></td>
</tr>
<tr>
<td>4.4 Example of the master branch.</td>
<td>47</td>
<td></td>
</tr>
<tr>
<td>4.5 Example of the choosing directory.</td>
<td>48</td>
<td></td>
</tr>
<tr>
<td>4.6 Example of the import projects.</td>
<td>48</td>
<td></td>
</tr>
<tr>
<td>4.7 Example of all the projects which has been import.</td>
<td>49</td>
<td></td>
</tr>
<tr>
<td>4.8 Example of the plugin of Archi in Eclipse.</td>
<td>49</td>
<td></td>
</tr>
<tr>
<td>4.9 New elements in core of archi</td>
<td>51</td>
<td></td>
</tr>
<tr>
<td>4.10 Image of the new element add to the Archi</td>
<td>53</td>
<td></td>
</tr>
<tr>
<td>4.11 Image of the new element into the palette of Archi</td>
<td>63</td>
<td></td>
</tr>
<tr>
<td>4.12 Final design of element router</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>4.13 Final design of element switch</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>4.14 Final design of element interface switch</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td>4.15 Final design of element interface router</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td>4.16 Final design of element routing table</td>
<td>66</td>
<td></td>
</tr>
<tr>
<td>5.1 Architecture of IXP</td>
<td>69</td>
<td></td>
</tr>
<tr>
<td>5.2 Infrastructure of an IXP model in Archi</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>5.3 Description of the case 1</td>
<td>71</td>
<td></td>
</tr>
<tr>
<td>5.4 The model of case 1 in Archi</td>
<td>72</td>
<td></td>
</tr>
<tr>
<td>5.5 Description of the case 2</td>
<td>73</td>
<td></td>
</tr>
<tr>
<td>5.6 The model of case 2 in Archi</td>
<td>73</td>
<td></td>
</tr>
<tr>
<td>5.7 Example of the impact in the process of adoption when the route fails or changed</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>5.8 Example of the impact in the attribution of scholarship when a route fails or changed</td>
<td>75</td>
<td></td>
</tr>
<tr>
<td>B.1 Example 1</td>
<td>92</td>
<td></td>
</tr>
<tr>
<td>B.2 Example 2</td>
<td>92</td>
<td></td>
</tr>
</tbody>
</table>
List of Tables

2.1 Example of a Profile for a modelling program [6] .................................................. 28
2.2 An example of a profile for specialisation of concepts on ArchiMate [6] .................. 28
3.1 The list of elements proposed for adding in ArchiMate model ............................... 35
4.1 Representation of the relationship in Archi ............................................................. 51
Listings

2.1 Routing Information Protocol (RIP) in Yet Another Next Generation (YANG) language . . . 23
2.2 The NDL code for the two routers [4] . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25
4.1 Adding new element to the list of technology layer . . . . . . . . . . . . . . . . . . . . . 52
4.2 Code for adding images . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 53
4.3 Creating the Graphical Modeling Framework (GEF) of the element interface router . . . 53
4.4 Creating the GEF of the element interface routing table . . . . . . . . . . . . . . . . . . . 55
4.5 Creating the GEF of the element for router . . . . . . . . . . . . . . . . . . . . . . . . . . . 56
4.6 Creating the user interface for interface router. . . . . . . . . . . . . . . . . . . . . . . . . 57
4.7 Creating the user interface for interface switch. . . . . . . . . . . . . . . . . . . . . . . . . 58
4.8 Creating the user interface for router. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 59
4.9 Creating the user interface for switch. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 60
4.10 The Extensible Markup Language (XML) file of the new elements add to the Archi . . 61
4.11 The XML file of the new elements add to the Archi . . . . . . . . . . . . . . . . . . . . . 62
## Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGIS</td>
<td>Apex Global Information Services</td>
</tr>
<tr>
<td>AMA</td>
<td>Agency for Administrative Modernisation</td>
</tr>
<tr>
<td>Ameritech</td>
<td>American Information Technologies Corporation</td>
</tr>
<tr>
<td>AMS-IX</td>
<td>Amsterdam Internet Exchange</td>
</tr>
<tr>
<td>AR Telecom</td>
<td>Acesso e Redes de Telecomunicações, S.A.</td>
</tr>
<tr>
<td>AS</td>
<td>Autonomous Systems</td>
</tr>
<tr>
<td>ASN</td>
<td>Autonomous System Number</td>
</tr>
<tr>
<td>BGP</td>
<td>Border Gateway Protocol</td>
</tr>
<tr>
<td>EA</td>
<td>Enterprise Architecture</td>
</tr>
<tr>
<td>EGP</td>
<td>Exterior Gateway Protocol</td>
</tr>
<tr>
<td>EJBs</td>
<td>Enterprise JavaBeans</td>
</tr>
<tr>
<td>eBGP</td>
<td>external BGP</td>
</tr>
<tr>
<td>EMF</td>
<td>Eclipse Modelling Framework</td>
</tr>
<tr>
<td>ESPAP</td>
<td>Entidade de Serviços Partilhados da Administração Pública</td>
</tr>
<tr>
<td>DEN-ng</td>
<td>Directory Enabled Network next generation</td>
</tr>
<tr>
<td>DE-CIX</td>
<td>Deutscher Commercial Internet Exchange</td>
</tr>
<tr>
<td>DSRM</td>
<td>Design Science Research Methodology</td>
</tr>
<tr>
<td>FCCN</td>
<td>Fundação para a Computação Científica Nacional</td>
</tr>
<tr>
<td>GigaPIX</td>
<td>Gigabit Portuguese Internet eXchange</td>
</tr>
<tr>
<td>Acronym</td>
<td>Full Form</td>
</tr>
<tr>
<td>---------</td>
<td>-----------</td>
</tr>
<tr>
<td>GEF</td>
<td>Graphical Modeling Framework</td>
</tr>
<tr>
<td>InP</td>
<td>Infrastructure Provider</td>
</tr>
<tr>
<td>iBGP</td>
<td>internal BGP</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IPV4</td>
<td>Internet Protocol Version 4</td>
</tr>
<tr>
<td>IPV6</td>
<td>Internet Protocol Version 6</td>
</tr>
<tr>
<td>IXP</td>
<td>Internet eXchange Point</td>
</tr>
<tr>
<td>ISPs</td>
<td>Internet Service Providers</td>
</tr>
<tr>
<td>JEE</td>
<td>Java Enterprise Edition</td>
</tr>
<tr>
<td>JDK</td>
<td>Java Development Kit</td>
</tr>
<tr>
<td>LINX</td>
<td>London Internet Exchange</td>
</tr>
<tr>
<td>NSFNET</td>
<td>National Science Foundation Network</td>
</tr>
<tr>
<td>NSF</td>
<td>National Science Foundation</td>
</tr>
<tr>
<td>NSP</td>
<td>Network Service Providers</td>
</tr>
<tr>
<td>NAP</td>
<td>Network Access Point</td>
</tr>
<tr>
<td>NAPs</td>
<td>Network Access Points</td>
</tr>
<tr>
<td>MAC</td>
<td>Media Access Control</td>
</tr>
<tr>
<td>NETCONF</td>
<td>Network configuration Protocol</td>
</tr>
<tr>
<td>NDL</td>
<td>Network Description language</td>
</tr>
<tr>
<td>MAE-EAST</td>
<td>Metropolitan Area Exchange/Ethernet East</td>
</tr>
<tr>
<td>MEO</td>
<td>Serviços de Comunicações e Multimédia</td>
</tr>
<tr>
<td>MOF</td>
<td>Meta Object Facility</td>
</tr>
<tr>
<td>Megaf</td>
<td>MEGa-modeling Architecture Frameworks</td>
</tr>
<tr>
<td>PacBell</td>
<td>Pacific Bell Telephone Company</td>
</tr>
<tr>
<td>PCH</td>
<td>Packet Clearing House</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Full Form</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
</tr>
<tr>
<td>PTT</td>
<td>Ponto de Troca de Tráfego</td>
</tr>
<tr>
<td>RCP</td>
<td>Remote Procedure Call</td>
</tr>
<tr>
<td>RFD</td>
<td>Resource Description Framework</td>
</tr>
<tr>
<td>RIP</td>
<td>Routing Information Protocol</td>
</tr>
<tr>
<td>rSPTIC</td>
<td>Rede Operacional de Serviços Partilhados TIC</td>
</tr>
<tr>
<td>RA</td>
<td>Routing Arbiter</td>
</tr>
<tr>
<td>RCP</td>
<td>Eclipse Rich Client Platform</td>
</tr>
<tr>
<td>SLS</td>
<td>Service Level Specification</td>
</tr>
<tr>
<td>SysML</td>
<td>System Modeling Language</td>
</tr>
<tr>
<td>TCP</td>
<td>Transport Control Protocol</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
<tr>
<td>VLAN</td>
<td>Virtual Layer Access Network</td>
</tr>
<tr>
<td>VN-SLA</td>
<td>Virtual Network Specification</td>
</tr>
<tr>
<td>VPN</td>
<td>Virtual Network Provider</td>
</tr>
<tr>
<td>WAN</td>
<td>Wide Area Network</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
<tr>
<td>YANG</td>
<td>Yet Another Next Generation</td>
</tr>
</tbody>
</table>
Introduction

Contents

1.1 Context .................................................. 3
1.2 Objectives ............................................... 4
1.3 Thesis Contributions ................................. 4
1.4 Definition of the Problem ......................... 5
1.5 Research Methodology ......................... 5
1.6 Outline of the Document ................... 6
1.1 Context

The web content has developed gradually over the years from a static text web-page to an interactive web-pages with high definition multimedia requiring a very high bandwidth for processing and transmitting. Therefore, it is vital to improve the delay time of these systems to accommodate high bandwidth usage. With the constant evolution in internet, the organisation wants a different solution to exchange information, without the help of intermediary operators to provide the various service to avoid the expensive services and higher latency in the packages delivered as the data packages has to pass through several networks before it reaches its destination. This situation made companies wants to have a single point where all the members can connect according to the rules and protocols established within themselves. The concept of connecting and establishing a relation between a set of Internet Service Providers (ISPs), university, content providers, and other types of companies to share information from a physical point is called Internet eXchange Point (IXP). An IXP provides an infrastructure where any member organisation can connect and share information to reduce the interconnection cost and expand the access network [7]. Over the last years, the IXP has been increasing in Europe significantly, representing an important role in Internet Backbone management. The IXP progressed and flourished rapidly because it allows their customers to use the services from any router server in the access network for free and with lower latency.

One of the main requirement in the design, implementation and operation of computing infrastructure such as IXP is the information distribution model [8], which is capable of capturing the requirements of physical infrastructure and split up the information of the application from the operating systems. The main advantage of this kind of information model is being able to represent how these different types of data and objects can relate to each other in a single consistent way without being influenced by the proficiency of repository [9]. There are number of Enterprise Architecture (EA) frameworks (such as ArchiMate, Zachman, TOGAF and etc) existing today which is used for representing and modelling the information. Due to the complexity of how the information systems are interconnected and constant evolution of different technologies, there is always a need for constant update in EA frameworks to catch up with the trend. In order to do so, the organisation has to resort to the addition of extensions to achieve their scopes. To constantly search for a new solution to improve the network performance and reduce the cost, they must have a framework which can represent the complexity of these systems for a better integration and management. Today, there are EA that can be used for modelling, management, integration and implementation of information, however it does not allow much diversity in relation to the implementation of the technological network infrastructure to support the process of a business. The techniques of modelling used by these EA, focuses on the business, product and application of the organisation. In a EA, new elements can be added without corrupting the integrity of existing concepts through the profiles or the specialisation of the concepts that are already existing. A profile is a structure
that can be defined in an independent application with new elements and integrated into architecture when there is a need for adapting the new elements [10]. The cost for adding profiles is relatively low in comparison with the acquisition of new architectures.

1.2 Objectives

The nature and purpose of this research is to find a way to improve the meta-model of the EA modelling language to implement IXP in an abstract way and help in the future implementations of network. Implementing the IXP in EA can allow the alignment between business strategies and the network provider. It also allows the communication between the different aspects of an organisation, from business process to the applications that support its and the technologies behind these services. This work will conduct surveys on different layers of network to understand how the modelling of information exchanged in an IXP is done without corrupting the integrity of the EA. After carrying out a thorough survey and literature review on the existing attributes of this kind of network, it is important to understand if these attributes are enough for the implementations or if it requires a new concepts to deploy the network abstraction for the development of this project.

1.3 Thesis Contributions

The main contributions of this work are:

- Adopting the selected EA ArchiMate framework and improve the modelling of network by adding new elements to the framework.

- Designing and adding the following new elements and its attributes for the Technology layer of ArchiMate.
  
  i. Router- Adding as attributes name, Internet Protocol (IP) address.
  ii. Switch- Adding as attributes name, Media Access Control (MAC) address
  iii. Interface Router- Adding as attribute name, system table, protocol.
  iv. Interface Switch - Adding as attribute, name, system table, protocol.
  v. Routing Table - Adding as attribute, name, a table with all the information for send a packet to the final destination.

- Implementation of new meta-model and optimisation of the exiting meta-model for modelling language, ArchiMate.
1.4 Definition of the Problem

This section presents the problems associated with the deployment of this project. To proceed with the implementation of the IXP in the ArchiMate, it is necessary to verify all the domains for the representation. To address the challenges imposed on this work, it is important to comprehend if it is possible to implement the IXP in an EA framework using a given sets of information? Hence, this work will lay out and construct the concepts in ArchiMate to formulate and frame the specialisation of all these concepts, and then deploy the chosen specialisation of the concepts to the framework. In order to specialise these concepts, following steps must be taken into considerations:

(i) Identifying the concepts presented in EA in order to understand which attributes exist and which ones can be added?

(ii) How to add these new attributes to an existing EA?

(iii) How the exiting meta-models can be updated with new attributes without corrupting the integrity of the EA?

(iv) How to maintain the consistency of the meta-models of modelling language?

1.5 Research Methodology

The objective of this project is do the modelling of elements in IXP by going through a set of steps, from the survey of the elements in existing EA to the addition of new elements, and to the implementation and evaluation of the final solution. The approach applied to this work is Design Science Research Methodology (DSRM) which focuses on the production of artifacts from a set problems to implement the solution. There are six phases of DSRM according to Ken Peffers et al. in [11] as shown below, and is illustrated in Figure 1.1.

- **Phase 1 - Identification of Problem and Motivation:** This phase identifies the problem of investigation and the proposed solution in order to solve the problem as described in Chapter 1.4.

- **Phase 2 - Definition of objectives for the solution:** In this phase, it will identify the prerequisite and other necessary requirements that should be followed to achieve the desired solution.

- **Phase 3 - Design and develop the artifacts:** In this phase, the solution is proposed and implemented to solve the problems defined in phase 1. For this project, the solution is proposed in Chapter 3 and the elements and attributes of the solution is implemented in Chapter 5.
• **Phase 4 - Demonstration:** This phase is the demonstration of the solution proposed. After the implementation of the new attributes, it illustrates a demo of the IXP abstraction deployed. The implementation of new attributes is described in the Chapter 4.

• **Phase 5 - Evaluation:** In this phase, the evaluation of the solution implemented for assurance. The evaluation methodology for this work is described in Chapter 5.

• **Phase 6 - Conclusion:** This phase draws conclusion of the solution implemented as well as describe the results obtained, as shown in Chapter 6.

![Figure 1.1: Phases of Design Science Research](image)

### 1.6 Outline of the Document

The remaining of this report is organised as follows: Chapter 1 presents the introduction of the work, the thesis problem and the research methodology used for the implementation of project. In chapter 2, it presents the proof of concept of the main solution of IXP with insights from the state-of-art IXP in the world. To complement the implementation of this project, it also explains the concepts of EA, list of framework used for modelling and the related work about adding profiles to it. Chapter 3 proposes the solution to formulate the attributes to solve the problems defined, and it also shows how it will implement the proposed solution. Then, the solution is implemented with the proposed attributes in Chapter 4 and the evaluation framework methodology for the new proposed architecture is presented in chapter 5. Finally, Chapter 6 draws the conclusion of this work by presenting the final remarks about the implementation and the main contributions.
## 2 Related Work

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Internet Exchange Point</td>
<td>9</td>
</tr>
<tr>
<td>2.2</td>
<td>Exchange Points In The World</td>
<td>10</td>
</tr>
<tr>
<td>2.3</td>
<td>Network Planning and Dimensioning</td>
<td>12</td>
</tr>
<tr>
<td>2.4</td>
<td>Enterprise Architecture</td>
<td>14</td>
</tr>
<tr>
<td>2.5</td>
<td>Enterprise Modelling Languages</td>
<td>14</td>
</tr>
<tr>
<td>2.6</td>
<td>Language Extension Mechanism</td>
<td>27</td>
</tr>
<tr>
<td>2.7</td>
<td>Proposed Extensions In Enterprise Architecture</td>
<td>29</td>
</tr>
</tbody>
</table>
This chapter reviews the concepts about IXP and describes the related work implemented in this area. This chapter starts by presenting the main definitions about the IXP and the existing projects in the world. It also describes the concepts about the EA and presents a list of attributes that can be used for implementation of the network projects. Additionally, it also presents some frameworks used for designing the network.

2.1 Internet Exchange Point

This section presents the main concepts about IXP. From 1987 until 1994, the operation for the backbone network of National Science Foundation Network (NSFNET) was responsible for National Science Foundation (NSF) in United States [12]. In the year of 1992, they started a plan to move all operations of the Internet core to a private sector. In this process of change, three main concepts such as Network Service Providers (NSP), Network Access Point (NAP) and Routing Arbiter (RA) came to light [2]. The NSP is responsible for all the operations of backbone core. The NAP will take care of the traffic exchanges between the several access point in the world. In order to establish the communication, RA provides the routing tables with all information regarding routes to the NAP, similar to the modern Router Servers. The concept of IXP was established in these years when the NAP was created to connect different NSP. Over the years, the NAP was reduced as it was maintained by a large company with private interest which led to the disconnection of some operators [2]. IXP provides an infrastructure to support the connection between several Autonomous Systems (AS) administrated by ISPs, where they can share information according to the rules and protocols established between them [13]. The AS interconnected to exchange points can share information by doing multilateral peering with all the members in connection, which typically is a router server that allows any member to use the Border Gateway Protocol (BGP) for receiving the information about the routes destinations in the connection. Also, it can establish bilateral peering to share information directly with each other. Most of the exchange point is implemented based on Virtual Layer Access Network (VLAN) with a configuration of port-based/tag-based to establish peering between the ISPs. In the port-based configuration, each port of switch will have a name to be used for connecting two different switches, and the tag-based configuration is a process where each switch is marked with a label, that contains the name or the number of the specific VLAN ID [14]. To summarise, every IXP should have the following requisites [15]:

- Each participant should have his own Autonomous System Number (ASN).
- Each participant needs to connect their Router to the IXP.
- The Router of each organisation has to be able to run the BGP because the traffic shared in the exchange point is done only by the protocol of routing mentioned above.
The participants of the IXP must abide by the terms and conditions imposed by the members.

Figure 2.1 shows the architecture of an IXP. The central point of an IXP is the switch where all the information goes through, and connects all the different AS of the different organisations. This makes the switch a focal point for exchanging all the traffic. The different members can share information between them by using Layer 2 as in VLAN Ethernet or Layer 3. For example, in Figure 2.1, if an organisation from AS 50 wants to have a private connection or a shared connection with others members, it has to fulfil all the rules established by the members of IXP and then only they can proceed to peer to the switch or with any other member.

Figure 2.1: Architecture of Internet Exchange Poin [2]

2.2 Exchange Points In The World

In the beginning, four Network Access Points (NAPs) were implemented, Metropolitan Area Exchange/Ethernet East (MAE-EAST), Sprint NAP, Pacific Bell Telephone Company (PacBell) NAP, and American Information Technologies Corporation (Ameritech) NAP. These NAP were supporting the commercial services which offered services for the Backbone (e.g, MCI.net, Sprint-link, Apex Global Information Services (AGIS)) and also maintained the network stability when the NSFNET was removed from the market. In the following year, the internet had grown to a higher level so these NAPs were replaced by the modern IXP [16]. Today, there are several projects of IXP implemented in different parts
of the world as shown in Figure 2.2. The image was taken from the Packet Clearing House (PCH), a non-profit institute for researching the routing and exchange of traffic, among other pertinent tasks for the operation of internet and economy. This organisation provides operational support and security to critical Internet infrastructure including IXP.

![Packet clearing House Exchange traffic point in the world](image)

However, in this project it will only specify some of the most successful exchange points, such as Deutscher Commercial Internet Exchange (DE-CIX) in Frankfurt, Amsterdam Internet Exchange (AMS-IX) in Amsterdam, London Internet Exchange (LINX) in London and the Gigabit Portuguese Internet eXchange (GigaPIX) in Portugal. The biggest IXP in the world was implemented in the year 1990 in Amsterdam, Holland for a non-profit purpose called AMS-IX. Currently, there are more than 750 networks connected to AMS-IX, such as content providers, television broadcasters, game companies and other types of networks. The DE-CIX in Frankfurt was implemented in the year of 1995 providing an infrastructure capable of connecting various types of ISPs like content providers, bandwidth content and other types of services. Nowadays, it has more than 800 clients in more than 60 countries including, Frankfurt, Hamburg, Munich, Dusseldorf, Palermo, Marseilles, Madrid, Istanbul, Dubai, New York and Dallas. The LINX in London has more than 600 members connected to it, in more than 66 different countries.

In Portugal, the most successful IXP is called GigaPIX that connects most of the major telecommunication companies and some of the universities, which leads to a total number of 25 members connecting to it. The members are organisations such as Vodafone, NOS communication, Serviços de Comunicações e Multimédia (MEO), Acesso e Redes de Telecomunicações, S.A. (AR Telecom), Network Telecommunication, Internet technologies in Angola and others. It was implemented by the Fundação para a Computação Científica Nacional (FCCN). GigaPIX has access points in Porto and Lisbon in Portugal. Currently, there is also another IXP being developed in
Portugal by Agency for Administrative Modernisation (AMA) and Entidade de Serviços Partilhados da Administração Pública (ESPAP) denominated as Rede Operacional de Serviços Partilhados TIC (rSPtic), which will connect most of the public services, Ministry of Education, Ministry of Finance, Ministry of Culture, Ministry of Economy, Ministry of Science and higher superior and others organisation.

2.3 Network Planning and Dimensioning

A network involves several concepts, from the equipment used to the protocols used in exchanging packets and the addresses for the identification of each equipment. This section describes the main concept about the network for understanding this project.

2.3.1 Border Gateway Protocol

In a sharing architecture of traffic, there is a set of AS in different places and to exchange traffic between them, it has to resort on BGP. The BGP is a protocol of routing used by the different AS to forward the shared packets. The first protocol was implemented in the year 1989 (RFC 1105) to replace the Exterior Gateway Protocol (EGP). Subsequently, BGP was updated to version BGP-4 in 1995, due to some problems encountered in previous versions. Currently, this version (BGP-4) is being used for exchanging the packages on Internet and it is also specified in RCF 4271 [17]. This protocol uses the algorithm distance vector to exchange information about the paths between the ISPs. The BGP uses the Transport Control Protocol (TCP) in the port 179 to exchange the information of routing. The BGP in an exchange traffic point is responsible for the following steps:

(i) The participants chooses the routes to where they want to send the information, routes about the ISPs that they want to connect.

(ii) They also apply the filters about which information to receive about the different routes.

Two routers that initiate communication between them are called neighbours or peers and a session created between them can be external BGP (eBGP) or internal BGP (iBGP). The eBGP session is created by the peers of different AS to exchange routing updates and iBGP session by the peers of the same AS to exchange. The main difference between them is how the router sends and receives the information, and also how the information arrives to them [18]. The Figure 2.3 is an example of types of session created between the AS. For example, if two AS in the network wants to connect with each other, then there are four types of messages that they can share and exchange as described below [19]:

(i) **Open message**: It opens a session between the peers, and this is the first message sent out after the connection is established.
(ii) **Update message**: It is used for informing the updates of the routes of other systems using BGP, and allowing the routers to maintain the consistency in the network.

(iii) **Notification message**: It is sent when there is a problem in the network or to cancel the active session.

(iv) **Keep-Alive message**: The routers have to send periodically the keep-alive message to maintain the connection with neighbours.

![Diagram of eBGP and iBGP between router AS](image)

**Figure 2.3**: Example of eBGP and iBGP between router AS [3]

### 2.3.2 Network Elements

The list below presents the concepts used in the network infrastructure which will be taken into consideration for the solution proposed in this project [20] [21]:

- **Domains**: It corresponds to an administrative domain used to divide the entities of topology into administrative groupings. In a standardised topology, all the nodes, paths, networks and bidirectional connection for the specific domain will be defined inside the domain element for that domain.

- **Nodes**: A element of a network which describes the entities like host and network devices.

- **Links**: It is an element which is used to describe the connection between two nodes. It may correspond to a physical connection such as Ethernet link or can be a Virtual Network Provider (VPN) or TCP connection.

- **Ports**: It is a component used to label the point of connection between a node and a link. This type of elements can be used to portray the Ethernet interfaces, listening to TCP and UDP socket.

- **Services**: They are used to identify the services available in the network.

- **Paths**: Includes a set of links, ports, nodes, domain to represent a network topology.

- **Networks**: Its a set of components used to represent the relation between a set of elements in the network topology. As an example, one element of a network can be the VLAN.
2.4 Enterprise Architecture

According to M.Lankhors [10], an EA is a group of principles, methods and models that are used for the conception and realisation of an organisational structure of a company in their process of business, information systems and infrastructure. This allows to capture the requirements of an organisation, which will be used to model the process of technological structure and the business process of an organisation. The main purpose of an EA is to be able to adapt the organisation to its own requirements [22]. With the constant evolution of globalisation, the rise of various business, the introduction of new technologies, the emergence of business processes, the EA confront with the need for the variety of new process to follow this rapid growth [23]. An EA is divided into the following architectures [24]:

![Layers of an Enterprise Architecture](image)

Figure 2.4: Layers of an Enterprise Architecture

Each one of these layers focuses on a specific area. The Strategy Architecture relates to the goal and objectives of an organisation. The Business Architecture gives support to the strategies and structure of an organisation. The Informational Architecture identifies what type of data will give support to the development of the business. The Application Architecture gives support to the essential applications to support the organisations business. The last, Technological Architecture will support the technologies which will be used to develops the different applications.

2.5 Enterprise Modelling Languages

The aim of this section is to describe different types of frameworks used for modelling the principles, business, terminologies and processes of an organisation. The main focus of this section is to explain the structure of the technology layer about the follow frameworks.
2.5.1 ArchiMate

ArchiMate is an open and independent EA modelling language to describe, analyse and visualise the relationships between architecture domains. The ArchiMate language allows representing the relations and dependency of an EA in a simplest way to integrate the process of an organisation. It also provides possibility of adding new concepts on an existing concepts or the specialisation of the existing concepts.

Figure 2.5 illustrates a process implemented by an organisation in the three layers of framework ArchiMate [25]. Each layer of the framework focuses on specific function which is explained in detail in the next section. The different colours distinguish the three layers of ArchiMate. The yellow colour represent the business layer, the blue colour represent the elements from application layer and the green ones are the elements from the technology layer.

The three layer of ArchiMate [10] are listed below:

(i) Business Layer: It offers the products and services to external customers.
2.5.1. A Business Layer

The business layer uses following concepts to model business processes of an organisation [10]:

(i) Business Service: A functionality which will increase value to the environment, independent of the way that this functionality works internally.

(ii) Business Process: The work flow of what activities to do in order to obtain the final result.

(iii) Business Interfaces: The physical location to access all the services available.

(iv) Business Function: This element offers a functionality that may be used for a business process.

(v) Business Actor: The entity which will request different services.

(vi) Business Interaction: It describes the behaviour element results from the collaboration of two business processes or more.

(vii) Business Event: It's an activity that can happen and influence business processes, functions, or interactions.

(viii) Business Object: The result of the behaviour of different processes.

These different concepts can be used to model a business process for any organisation. For example, a business service could be an Internet service connection or a cloud service and business process could be all the methods that a mobile site is planned until the end user. The business interfaces can be the Cloud or the Internet. Business Actor is any person who is requesting a service. A Business Interaction is an interaction between different services provided in the organisation like optimisation and planning of a mobile site.

2.5.1. B Application Layer

The Application Layer uses following concepts to model the process of any application in the business [10]:

(ii) Application Layer: Supports applications services that are ran by different application.

(iii) Technology Layer: Provides an infrastructure to run the application used by the computer and system software.

The list of concepts within each layer of framework used for modelling the process of an organisation in ArchiMate is described in following sections [10].
(i) Application Function: It is the result of any component or application required by the services. It describes the internal behaviour of an application component.

(ii) Application Service: It is a group of functionalities available to be used by the components in a process of business.

(iii) Application Interaction: This function is a result of a collaboration between two services. It describes the behaviours of the components in a collaboration.

(iv) Application Interface: It defines a set of operations and events that are provided by the component or a interface that will be required for another operation.

(v) Application Collaboration: It is a aggregation of two or more applications components that work together to produce a behaviour.

(vi) Data Object. A set of information that will be used for the automation of a process.

All the concepts mentioned above can be used by different applications developed by any organisation. The application function can be a policy of any company which can be use by an insurance policy. The application service can be seen as creation of a policy for a insurance company. A data object can be the application of insurance to produce all the policies.

2.5.1.C Technology Layer

The Technology layer use the following concepts to implement any infrastructure of a network to support different types of services deploy in the equipment [10].

(i) Node: It represents an entity of a technological layer.

(ii) Device: They are physical components, where the artifacts can be implemented to process the data.

(iii) Infrastructure Interface: It is defined as the logical localisation where the services are available by a component and it can be accessed by other components from the application layer.

(iv) Infrastructure Software: A software environment for the implementation of a specific type of components and data objects that are implemented in it in the form of artifacts.

(v) Infrastructure Service: A set of functionality well defined provided by the technological architecture as the Application Functions from the Application layer.

(vi) Artifact: A part of the physical information that is used or produced in the development of software or by the implementation and operation of a system.
(vii) Communication Path: It connects two nodes in order to share information.

(viii) Network: Physical communication between two devices or more.

The network element represents a physical connection between several AS. A device is typically used to model system hardware like a computer, mainframe, and routers. Infrastructure Service can be classified by three types of services, processing, storage and access to the communication services. The artifacts are data or file which has information about the application that will be modelled. The Infrastructure Software refers to a type of Java Enterprise Edition (JEE) server-type execution environment. An Infrastructure Interface specifies the type of protocols that a device needs to follow. A Communication Path is a path where the information exchange between two devices is transfer, like packets TCP/IP.

Figure 2.6 shows the elements from technology layer used for modelling a network infrastructure. In this scheme, it represents two devices interconnected using an access network layer. The series-Mainframe has a database, where it keeps all the information regarding an organisation. The Sun Blade provides a server called IPlanet APP Server to access the application Financial application Enterprise JavaBeans (EJBs).

![Figure 2.6: Example of a network implemented in ArchiMate](image)

2.5.1.D ArchiMate Meta-Model

Figure 2.7 it represents the meta-model of three layers of respective ArchiMate [26]. The purpose of this meta-model is to show an overview of the concepts and how is the elements are connected through the different concepts of relationship in ArchiMate. This figure does not show all the permitted relationship because every element would have the composition, aggregation, and specialisation relationship if they are of same type [26]. To achieve the goal of this project, the element node, infrastructure interface and communication path it will be used for the specialisation of new concepts described in chapter 3.
Figure 2.7: Example of ArchiMate Meta-model

Figure 2.8: Example of Technology Layer meta-model
Figure 2.8 presents the meta model of technology layer of ArchiMate. It shows all the relation between the concepts and relationships of the tool ArchiMate.

---

**Realisation:** This indicates that an entity play a critical role in the creation, achievement, sustenance, or operation of a more abstract entity.

**Serving:** This relationship describes how the services or interfaces offered by a behaviour or active structure element serves entities in their environment. An active structure represents an entity that is capable of performing behaviour. It can be divided into internal and external [28]. Example of internal can be a device, nodes and applications, whereas external can be the behaviour of interface.

**Access:** This indicates that a process, function, interaction, service or event does something with a passive structure element. A passive structure element is a structural element that cannot perform a behaviour [28].

**Triggering:** It is used to model the temporal or causal precedence of behaviour elements in a process. It describes a temporal or casual relationship between elements.

**Flow:** This relationship is used to model the flow of a information, goods between behaviour elements.

**Composition:** It indicates that an element consists of one or more elements. This relationship is always allowed between the elements of the same type.

**Aggregation:** This indicates that an element groups a number of other elements and it always allowed between two instances of same type.

**Assignment:** This expresses the allocation of responsibility and performance of behaviour or execution. It links assignment of business actors with business roles that are fulfilled by them.

**Association:** It is used to modulate an unspecified relationship or one that is represented by another ArchiMate relationship. In can be used for drawing a first high-level model where relationships are initially denoted in a generic way, and later refined to illustrate a more specific relationship types.

**Specialisation:** This relationship has been inspired by the generalisation of relationship in Unified
Modeling Language (UML) class diagram. The specialisation is when an element is divided in other elements of same type but with some different concepts. It is always applied between two instance of same type.

2.5.2 System Modelling Language

System Modeling Language (SysML) is a modelling language that supports specification, analysis, design, verification and validation of a set of systems [29]. These systems may include hardware, software, information, process and other types of services. This language is used as a reference to the definition of the concepts from UML [30]. UML is a modelling language for visualising, specifying and documenting artifacts of a software system [31]. The SysML is subdivided into three types [30] as shown below:

- Diagrams to capture the requirements of the systems and the physical limitations;
- Diagrams that describe the system structure;
- Diagrams that describe the behaviour of the system.

Figure 2.10 shows the four pillars (requirements, structure, behaviours and parametric) which represents SysML. The system structure is represented by block diagram, internal diagram and package diagram as explained in the list below [32],

i. A block definition diagram portray the system hierarchy and system component classifications.

ii. The internal block diagram narrate the internal structure of a system in terms of parts, ports and connectors.

iii. The package diagram is used to organise the model.

The behaviour diagrams are divided into activity diagram, sequence diagram, state machine and use case diagram as explained in list below [32],

i. Activity diagram represents the flow of data and control between activities.

ii. Sequence diagram represents the interaction between collaborating parts of a system.

iii. State machine diagram describes the state transitions and actions that a system or its parts performs in response to events.

iv. A use-case diagram provides a high-level description of the system functionality.
The requirement diagram captures requirements hierarchies and the derivation, satisfaction, verification and refinement relationships while the parametric diagram represents the constraints on system parameter values such as performance, reliability and mass properties to support engineering analysis [32]. Each one of these pillars is used to capture the main requirements for modelling. The implementation of a network should consider all the four pillars to describe how the devices are connected and what are the behaviours expected when they are trying to communicate with each other.

![Diagram](image)

**Figure 2.10:** Pillars of SysML

### 2.5.3 Yet Another Next Generation

The Yet Another Next Generation (YANG) is a data modelling language used for configuration, identification and state data manipulated by the Network configuration Protocol (NETCONF) [33]. The NETCONF provides a framework which allows to install, manipulate and erase settings of a device. The data configuration follows a tree directory and can be complex type list or joins. At YANG framework, the configuration of the data is partitioned into modules with multiple hierarchies. These hierarchies contains a set of nodes corresponding to the data settings, data state, Eclipse Rich Client Platform (RCP), and notifications [34]. This framework is not the best one to describe all the hardware, and devices in the network infrastructure, but it is very useful to describe how several protocols are used for communication between components. Figure 2.1 is an example of Routing Information Protocol (RIP) modelling in YANG [34]. It describes the information regarding the version of routing, network mask, and it also identifies the IP address if it is an Internet Protocol Version 4 (IPV4) or an Internet Protocol Version 6 (IPV6) for a simple RIP configuration. Lines 2-5 identifies the IP address of device and the corresponding subnet mask, line 6-8 is another IP address for the protocol. The lines 2-8 is the Extensible Markup Language (XML) of the corresponds RIP. Lines 12-17 is the description of the protocol which is been
used. Lines 18-22 is the type of IP address used for change the packet. Lines 24-26 describe the list of neighbours and what type of address is going to use.

Listing 2.1: RIP in YANG language

```yang
<rip>
  <version>1</version>
  <network>
    <address>192.168.1.0</address><mask>24</mask>
  </network>
  <network>
    <address>192.168.10.0</address><mask>24</mask>
  </network>
</rip>

Modelling router{
  namespace "http://cdm.depaul.edu/ns/router";
  import ietf-inet-types {
    prefix inet
  }
  container rip{
    leaf version
      type int8 {range "1..2";}
      default 1;
  }
  list network{
    key "address mask"
    leaf address {type inet:IPV4-address; }
    leaf mask {type int16 {range "0..31";}}
  }
  leaf-list neighbor{
    type inet:ipv4-address
  }
}

2.5.4 Network Description Language

A Network Description language (NDL) was implemented in the year 2000 by the University of Amsterdam [8]. NDL adopted the semantics of the page web for their schemes, particularly the Resource Description Framework (RFD). A RFD is a standard model for data interchange on the web [35]. The
NDL [36] has schemes (represented in UML) in a simple way to implement characteristics of a network. The aspects which represent a network are characterised by several aspects such as network topology, technology layers, device configurations, capabilities and aggregations between the devices which are used for modelling different aspects of network infrastructure [36]. The list below presents the four classes (location, device, interface and link) used by the framework NDL [4]:

- **Location**: Space where the devices are located. For example, it can be a server, or a cloud computing.

- **Device**: Any machine that is connected to a network. For example, a computer, tablets, Desktops, Laptop, Router, Switch, Server.

- **Interface**: It is the connection between devices and the rest of the network. For example, it can be the connection between two Routers to exchange the traffic.

- **Link**: It is the connection between two interfaces.

The following list presents the eight propierties that are related to the four classes presented above [4]. This relation allows to represent the description of network, cables, the interface they are connect and the capacity of a link.

- **Name**: It is the designation name of the interface.

- **Description**: A property that allows to include information that is visible to users.

- **LocatedAt**: The relation between device and localisation.

- **HasInterface**: Defines the relationship between device and interface.

- **ConnectedTo**: It is used to describe the physical connection interfaces or between a link.

- **Capacity**: Sets the bandwidth capacity of an interface or a link.

- **EncodingType**: Defines what type of encoding is used in an interface or a link.

- **EncodingLabel**: It shows information about the coding used in a link or a interface.

Figure 2.11 shows the four classes and the eight propierties. These four classes and eight propierties will be used together to represent a network. For example the class device can have the propierties like name, location and the corresponding interface (name, locateAt, hasInterface). The interface can use the propierties like name, connectedTo and capacity for the information regarding is name and which interface is connected.
Figure 2.11 shows an example of how connections between the two routers are established. The router with name tmd3.uva.netherlight.nl and interface 501/1 is requesting to make a connection with router which the name is tdm4.uva.netherlight.nl and with interface 5/1. To make the connection, the following configuration as show in listing 2.2 the needs to be done.

Figure 2.12 is a XML coding of how the configuration is done by two routers when the router with name "tdm3.amsterdam1.netherlight.net" is trying to connect to router named "tdm4.amsterdam1.netherlight.net". Lines 4-6 define the location of the device(Netherlight). Lines 7-11 is the information about the router named tdm3.amsterdam1.netherlight.net. Line 8 specifies that the device has an interface named 501/1 which is described in the lines 12-17. Line 15 defines that the capacity of a device is $1.2 \cdot 10^9$ bytes per second. Line 18-22 shows the information about the router named tdm4.amsterdam1.netherlight.net. Line 21 specifies that the device has an interface named 5/1 which is described in line 23-28. Finally, the line 26 also define the capacity of device.

Listing 2.2: The NDL code for the two routers [4].

```xml
<?xml version="1.0" encoding="UTF-8"?>
<rfd:RFD xmlns:rfd="http://www.w3.org/1999/02/22-rfd-syntax-ns#"
xsln:dnl="http://www.science.uva.nl/research/sne/ndl#">
<ndl:Location rdf:about="Netherlight">
  <ndl:name>Netherlight Optical Exchange</ndl:name>
</ndl:Location>
<ndl:Device rdf:about="#tdm3.amsterdam1.netherlight.net">
  <ndl:name>tdm3.amsterdam1.netherlight.net</ndl:name>
</ndl:Device>
```
2.5.5 Directory Enabled Network next generation

The Directory Enabled Network next generation (DEN-ng) is an object oriented information model which provides an architecture for representing information management such as devices, services, and users by taking into account the policy and the context which is integrated [36]. The description of managed entities is done by the different views such as business, systems, implementation, deployment and the information in the model is organised by the use of a unique class subdivided into three sub-classes (Entity, Value and Meta Data) [36]. The DEN-ng focuses on how the main concepts of network infrastructure such device, network and application can be represented. One of the main advantages of this model for the management of configuration is the outstanding model provide by framework for describing a network of multiple-service [37].

2.5.6 Comparative Analysis

The framework ArchiMate is a simple enterprise architecture, where the concepts are well structured. It allows the integration of new concepts, without changing the architecture semantics and is also a widely
used framework to design the business structures of a certain organisation. The framework SysML was implemented based on the language of UML, to capture the requirements of an organisation relative to their infrastructure of network and applications that are being processed in different machines. The framework YANG is mainly used to represent the interface of the network and the different protocol of routing used to exchange traffic between the devices. Whereas, NDL identifies the basic elements of a network such as devices, links, etc, and describes how the information is shared between the layer of the network also adding the concept of adaption [8].

2.6 Language Extension Mechanism

The extension mechanism refers to the concept of adding more elements to the modelling language without damaging the semantic of the framework. According to the techniques of UML of adding extensions, it can be described into heavyweight and lightweight ways. The heavyweight is implemented through the mechanisms of Meta Object Facility (MOF), which allows to use meta-models of the modelling framework. This way of adding the extension allows modification in existing meta-models and also to create new meta-models without any restriction. The method heavyweight is more flexible, allowing the alteration of the concepts without any restriction which can affect the standardisation of the enterprise architecture. The method of adding an extension in the lightweight does not allow the modification on the existent meta-models, however, it will allow to adapting them [38], which will lead to the so-called profile. The method lightweight is more restricted to the changes in the enterprise architecture, the user can change the meta-models according to their necessity but maintaining the standardisation of meta-models [39]. In this project the framework used for the development of the proposed solution is ArchiMate. There are two ways of adding concepts in ArchiMate [6].

1. Adding attributes to the elements and relationships of ArchiMate:

This method of adding elements is implemented by the addition of profiles. A profile is a structure of data that can be defined apart from the modelling language and can be integrated with the concepts and the relation of modelling language ArchiMate. The profiles allows to redefine a language in a strict way without contradicting the semantics of the framework [40]. The profiles can be distinguished by the pre-defined profiles and user-defined profiles [6]. A pre-defined profile has a defined attribute and can be implemented in any tool that supports ArchiMate language. A user-defined profile is when through a profiling language the user can add new profiles in order to extend the concepts or adding the relationships with a set of new attributes. Table 2.1 is an example of a profile with the method of adding new attributes to an EA. In the table 2.1 there are three type of profiles proposed. Each “Serving” relationship may have weight for indicated the average of number uses. Each “service” which could business, application or technology may
have a fixed and variable cost. The "structure element" which could be a device may have fixed or variable cost and a capacity [6].

Table 2.1: Example of a Profile for a modelling program [6]

<table>
<thead>
<tr>
<th>“Serving” Profile</th>
<th>“Service” Profile</th>
<th>“Structure Element” Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Type</td>
</tr>
<tr>
<td>Weight</td>
<td>Real</td>
<td>Fixed cost</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Currency</td>
</tr>
<tr>
<td>Variable cost</td>
<td>Currency</td>
<td>Variable cost</td>
</tr>
<tr>
<td>Service Time</td>
<td>Time</td>
<td>Capacity</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Integer</td>
</tr>
</tbody>
</table>

2. Specialisation of the existing elements and relations:

It is a powerful way of adding new attributes to the elements that are already on the framework. If an existing element is not able to represent the process that you want to designing it is possible to create a subset of the existing ones with new attributes. Table 2.2 presents an example of how the existing concepts can be specialised with new attributes according to the requirement of a user. Table 2.2 exemplifies the specialisation of five concepts of model framework ArchiMate, Node, Device, Network and Technology Service. The concepts node can be distinguished in logical component like packet TCP/User Datagram Protocol (UDP) or physical component like router that defines how these packages will be fill and shared. The concept of device can be mobile device or computer. The concept network can be Wireless Local Area Network (WLAN) OR Wide Area Network (WAN) which can be used to give coverage in a large geographical area The service Technology can be also divided in different storage type.

Table 2.2: An example of a profile for specialisation of concepts on ArchiMate [6].

<table>
<thead>
<tr>
<th>Parent concept</th>
<th>Specialised Concept</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>Logical Technology Component</td>
<td>An encapsulation of technology infrastructure of a particular product. A class of technology product.</td>
</tr>
<tr>
<td></td>
<td>Physical Technology Component</td>
<td>A specific technology infrastructure product or technology infrastructure product instance.</td>
</tr>
<tr>
<td>Device</td>
<td>Mobile Device</td>
<td>A portable device such as a smartphone or tablet.</td>
</tr>
<tr>
<td></td>
<td>Embedded Device</td>
<td>A computing device that is part of a piece of equipment.</td>
</tr>
<tr>
<td>Network</td>
<td>WiFi Network</td>
<td>Wireless Local Area Network (WLAN).</td>
</tr>
<tr>
<td></td>
<td>Wide Area Network</td>
<td>Along-range data communication network.</td>
</tr>
<tr>
<td>Technology Service</td>
<td>Processing Service</td>
<td>Service used for processing data by a node.</td>
</tr>
<tr>
<td></td>
<td>Storage Service</td>
<td>Service used for storing data on a node, typically offered by a database or file system.</td>
</tr>
<tr>
<td></td>
<td>Communication Service</td>
<td>Service used for transporting information (e.g., voice, data) between nodes.</td>
</tr>
</tbody>
</table>
2.7 Proposed Extensions In Enterprise Architecture

This section presents some of the projects implemented in the enterprise architecture by adding an extension to the concepts. It presents a critical analysis about the proposals described in this section.

Fajjari et al. in [41] proposed a Virtual Network Specification (VN-SLA) based scheme of XML to abstract the properties and relation of components in a network. The VN-SLA focuses on three distinct concepts, Party, Network Specification and Obligation. The Party describes the parties involved in the installation of a virtual network which includes the parameters, Infrastructure Provider (InP), VPN and clients. It is also possible to define which are the parties that will be the provider or consumer of resources. Network Specification describes the components of a virtual network, the association between them and the Service Level Specification (SLS) agreed by them. A Service Level Specification defines the application requirements. As for example, end-to-end delay, drop probability, latency and multi path bandwidth. The Obligation defines the responsibility in terms of guarantees and penalties of the parties involved. This principle also verifies if the services levels are guaranteed by the supplier or not, and to charge the entities involved if it does not verify.

Chiprianov et al. in [5] proposes a Telecom ArchiMate profile relying on a Meta-modelling approach. This method enables the addition of the Profiles in two ways by adding new attributes to the concepts of ArchiMate or by specialising the concepts that are already existing. The solution of Meta Modelling approach allows the use of the framework like Eclipse Modelling Framework (EMF) and Xpand to produce language-specific tools such as graphical editor and code generation. The proposed solution is based on the specialisation of the concepts from the technology layer of ArchiMate using as example the Core Network Subsystem. This solution was modelled for a audio conference at organisation, it shows how to participate in an audio conference and the network who supports this design. Figure 2.13 is an example of telecom extension for ArchiMate technology layer meta-model.
Figure 2.13: Example of a telecom extension for ArchiMate technology layer meta-model [5]

Hilliard et al. in [42] proposed the MEGa-modeling Architecture Frameworks (Megaf) which is a framework to implement EA. The solution Megaf was implemented based on models of mega-modelling techniques to promote the use of each element of a business architecture or the addition of the new elements. It considers the views, viewpoints, stakeholders and system concerning the first elements of the class mega-model, which will allow the engineer of software to define how to combine the elements or save it in order to produce the EA according to their expectations. The implementation of this solution was implemented based on the Eclipse plugin. Figure 2.15 shows how the elements are interconnected in the several layers. From a set of requirements specified by a organisation, the framework can model an EA which the entities want to design.
Quartel et al. in [43] proposed an extension that can be used in conjunction with ArchiMate called ARMOR. It implements a solution for modelling the goals, motivation and requirements of an EA and it also can be used in conjunction with others EA. This method extends the modulation of structure on ArchiMate by focusing on the concept of motivation from the point of view of the various companies, such as objectives and their intentions. In order to achieve this goal, it uses three concepts: Stakeholder domain, Principles domain, and Requirements domain. Stakeholder domain focuses on the interests of the companies including their concerns and evaluation of them. Principles domain identifies whether there is a need to define new strategies, missions, principles, and guidelines; however it does not know how to implement it. The Requirements domain models the goals, requirement and the expectation that may difficult the design of an EA. Figure 2.15 is the meta-model of the extension ARMOR.

2.7.1 Critical Analysis

As mentioned in section 2.7, there are some works which have been developed in the context of adding an extension to an EA. VN-SLA scheme specifies the resources associated with the virtual network, maintaining control over the defined agreements and relations between the various participants who are interconnected in a network. This paper focuses on the concept of virtualisation, where the application is separated from operational systems and it is implemented based on the language of the XML. The solution of Megaf reinforces the possibility of improving an EA which can meet the requirements of a
particular organisation. Although this solution reinforces the relationship between a set of elements representing modulation by adding new attributes or new rules, however they do not maintain the semantics of the architecture that is being changed. However the solution ARMOR in paper [43] proposed an extension which can be used in conjunction with the ArchiMate in order to improve its objectives in relation to the concerns of a particular entity. The concepts of ARMOR cannot be expressed as a new attribute or restriction for the different layers of the ArchiMate. This solution does not use the concept of adding the profiles whereby it does not allow the modification of concepts that already exist and the main focus of this solution is business layer. Lastly, the solution of Telecom extension for ArchiMate technology layer which focuses on the technology layer proposes a design for adding or change the concepts that is already existing. This solution allows adding new elements without changing the semantics of the others element. The fact that this solution was developed to alter the technological layer, its emphasis more with the requirements of this project as discussed in chapter 3.
3
Proposed Solution

Contents

3.1 New ArchiMate Elements Proposed ........................................... 35
3.2 New Meta-Model of Technology Layer in ArchiMate ......................... 36
3.3 New Detailed Meta-Model of ArchiMate ........................................ 39
As discussed in previous chapter, the EA and IXP are two conceptualised concepts in the market with a lot of opportunities to improve. From the research conducted in Figure 2, it was possible to identify that the concepts present in the enterprise modelling language are too abstract for describe the information about the network where a more detailed information about the system it is needed.

The purpose of this thesis is addition of new concepts to the modelling language ArchiMate through the specialisation of concepts to be able to model the IXP into more detailed way or others types of network provider. In our approach, the specialisation mechanism was used to add new concepts and attributes.

The motivation behind choosing of the ArchiMate for modelling the IXP was because it provides concepts for expresses, designing and validate the different aspects of an EA such as the business process, the applications for these processes and the network infrastructure supporting these applications. It also has a powerful way to add new attributes to the existing concepts and relation trough the method of specialisation or adding profiles without modifying the syntax or semantics of the model language. It reduces the complexity and proposes a model for a better understanding of enterprise, the interconnections and inter-dependency between the process, the people, the information and the EA systems [44].

### 3.1 New ArchiMate Elements Proposed

After validating the existing concepts, it is possible to conclude which concepts to add in the ArchiMate for modelling an IXP. The concepts added to this project are router, switch, interface switch, interface router and routing table as shown in the table 3.1.

<table>
<thead>
<tr>
<th>Attributes Added</th>
<th>Descriptions of concepts</th>
<th>ArchiMate Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Router</td>
<td>Represents one of entities used in the communication</td>
<td>![Router Icon]</td>
</tr>
<tr>
<td>Switch</td>
<td>Represents one of entities used in the communication</td>
<td>![Switch Icon]</td>
</tr>
<tr>
<td>Routing Table</td>
<td>The information about the different network destinations</td>
<td>![Routing Table Icon]</td>
</tr>
<tr>
<td>Switch Interface</td>
<td>The Information about the interface of Switch connected</td>
<td>![Switch Interface Icon]</td>
</tr>
<tr>
<td>Router Interface</td>
<td>The Information about the interface of Router connected</td>
<td>![Router Interface Icon]</td>
</tr>
</tbody>
</table>

The router is added because just with the element node it is not possible to identify the IP address of the devices. The switch is added because element node cannot identify which MAC address is
used to identify the device of different organisation. The element switch interface and element router interface was add to the ArchiMate because with the infrastructure interface is impossible to represent the active interfaces and the name of the protocol used for exchanging packets. The routing Table was add because with the element path is not possible to represent the list of routes and the IP address for where the router has to send the packets over the network. The addition of these attributes will distinguish the different AS, the IP addresses used to identify the equipment and the information used for information exchange between the organisation to connect to the IXP.

3.2 New Meta-Model of Technology Layer in ArchiMate

Figure 3.1 represents the meta-model of the technology layer of ArchiMate with the new elements proposed for this project. The concepts added are router, switch, interface switch, interface router and routing table.

![Figure 3.1: New proposed ArchiMate Technology model](image)

3.2.1 Router

It is the system used by the organisation for exchanging of packets with different entities involved in the IXP. It is very important as it is the main means for the entities involves in the exchange point has to
exchange the packets. The element router is the specialisation of element node. It will have the following attributes for distinguish the different organisation connected,

- Name: Name of the corresponding interface.
- IP Address: The identification of the router that will be used in the network for exchanging packets.

The notation of the concept router is represented in the Figure 3.2.

![Figure 3.2: Router Notation](image)

### 3.2.2 Switch

It is one of the main entities in the IXP. It will be responsible for exchanging the packets layer 2 or layer 3. Its the main point for where all the organisation has to connect and exchange traffic trough it. It will include following attributes for differentiate the different organisation connected. Switch is a specialisation of element node. The notation of the concept Switch is represented in the Figure 3.3.

![Figure 3.3: Switch Notation](image)

It will include following attributes:

- Name: Name of the switch interface.
- Mac Address: The corresponding identification of switch interface.

### 3.2.3 Routing Table

This entity contains all the information about the topology of the network which can be used for exchanging all the packets for different organisations to connect to the IXP. It will listing the routes for a particular destination. The element routing table is the specialisation of the element path and it will be used for the element interface router to verify the different the destination. The information that is going to stored are listed below.
- Name: Name of the corresponding Interface.

- Routing Table: Information about the destinations of packets to be sent, information about the gateway, net mask, IP destination and the Interface.

The notation of the concept routing table is represented in the Figure 3.4

![Routing Table Notation](image)

**Figure 3.4: Routing Table Notation**

### 3.2.4 Interface Switch

It represents the information about the different interface of switch and the respective IP which are connected with other systems. This element will have the following attributes. The interface switch is the specialisation and aggregation of new element technology interface and on the other hand it can be used by the new element switch.

- Name: Name of the system that are represented.

- System Table: Name of all interfaces and their respective MAC addresses.

- Protocol: Name of the respective protocol used for the interchanging packets.

The notation of the element interface switch is represented in the Figure 3.5

![Interface Switch Notation](image)

**Figure 3.5: Interface Switch Notation**

### 3.2.5 Interface Router

It represents the information about the Router and the respective IP which are connected with other systems. The interface router is the specialisation and aggregation of the element technology interface and it can be used by the new element router.

The element interface router will have the following attributes.

- Name: Name of the system that are represent.
- System Table: Name of all interfaces and their respective IP.

- Protocol: Name of the respective protocol used for the interchanging packets.

The notation of the element interface switch is represented in the Figure 3.6.

![Figure 3.6: Interface Router Notation](image)

The elements are connected by the different possibility of the relationship that they could have. The relationships are realisation, serving, access, triggering, flow, composition, aggregation, assignment, association and specialisation. The relationships are explained in chapter 2.

### 3.3 New Detailed Meta-Model of ArchiMate

Figure 3.7 is a detailed meta-model of the proposed concepts and how the elements are connected to each other. The definition of arrows connecting each element is explained in the chapter 2. The element switch and router are a specialisation of the element node. The element interface switch is a specialisation of element infrastructure interface and it can be used by the element switch.

The element interface router is a specialisation of the element infrastructure interface and it can be used by the element router.

The element routing table is the specialisation of the element communication path and it can be used by the element interface router.

The element device is the specialisation of the element node which means it can be used for any critical operation. The element device is also associated with the element network which means that a device can have access to the network and it is assigned to element system software which has the same relationship.

A node can contain one or more devices. The element node it composes by infrastructure software, and his result can be used by the node. The element node is also connect to Communication path for sharing information about the different network.

The communication network aggregates the system software, device, router and switch. A communication network realises the communication path.

The element artifact is used by node for a operation. The element node realises the element infrastructure interface.
The infrastructure service and infrastructure interface are join because the infrastructure interface may contain services which can be access by the infrastructure service. The element infrastructure of service can produces and uses the element artifact.

The infrastructure interface can have a functionality that result from the node which can be access by the element application functionality.

The application function is the behaviour of a element that are realises by the application service which will be available to the users. The application component encapsulates is behaviour and data in order to exposes them through application interface and it aggregates to the element application collaboration.

The application component has the same relationship as application function and the element business process.

The application interface is join with the application service because is where the the application service is available to the users and it also connect to the business service.

The application service expose a functionality of components to the environment and it will be accessed through the application interfaces and it is a part of element product.

The element business represent a sequence of business behaviour can access the business object. A business process may realise one or more business services and may use business services.

A business role may be assigned to a business process. A business role may be part of a business collaboration. A business service is a part of the product and it can be access by the business Service.

A business service may serve a business process, business function, and business interaction. A business process, business function, or business interaction may realise a business service.

A business service may access business objects. A business interface may be assigned to business services, which means that these services are exposed by the interface.

A business interface my be a part of business role or may serve the business role. A business collaboration may realise the business role.

A business actor which is responsibly for performing a specific behaviour my be assigned to a business role. A product which is a collection service my aggregate the application service, business service and a contract. A value may associate with the product. A business object my realise the data object. The representation can realise the business objects.
Figure 3.7: New Proposed ArchiMate Meta-Model
Implementation of Proposed Solution

Contents

4.1 Software Setup ......................................................... 45
4.2 Adding New Concepts to ArchiMate ............................... 50
This chapter 4 will discuss the tool set ups and implementation steps of adding the concepts that were proposed in chapter 3 using ArchiMate.

4.1 Software Setup

The deployment of the solution was done over the plugin of Archi tool. The plugin of the program is available in the Git repository. Git is a distributed version control system for tracking changes in a source code during the software development. Archi was built upon the RCP version 4.7 (Oxygen) and to implement any solution in Archi, following set-up steps must be carried out [45]:

**Install Java Development Kit (JDK):** The framework Eclipse runs on top of java so it is important to make that a correct version of Java is installed. Archi does not support the Java 9 yet, so it is necessary to install Java JDK 8.

**Install Eclipse:** For the implementation of this work, Eclipse will be the main framework. You have to install the correct version of Eclipse to use the plugin of Archi, which is Eclipse IDE version 4.7 (Oxygen). It is also necessary to download a package that includes Git clients, supported by Eclipse plugin and Rich Client Platform development.

**Setting up Eclipse:** After downloading the Eclipse package, it will create a folder on your computer. When you open the folder click on the executable file of eclipse and it will ask to select the work-space. After selecting the work-space, it will show the working window.

**Adding Additional Plugin:** If the Eclipse dos not include the Git client, install it within the Eclipse itself using following steps:

1. Choose Help - Install New software.
3. Wait for the dialog list to populate.
4. From the list under “Collaboration” select “Git integration for Eclipse.”
5. Click “Next” and Accept all terms to Finish the wizard.
6. The libraries should then be installed and you will need to restart Eclipse

Now that the Eclipse environment is all set, import the Archi codes to Eclipse environment using the following instructions [45]:

45
1. From within Eclipse choose "File-Import"

2. In the wizard select "Git - Projects from Git". Click "Next"

3. Select "Clone URI" and then click "Next"

**Figure 4.1:** Example of git properties.

**Figure 4.2:** Example of Clone URI.
4. Add the details for the repository’s URI (see above) and then click "Next".

![Figure 4.3: Example of repository URI.](image)

5. Make sure that the “master” branch is selected (ignore all other branches that you may see) and then click "Next".

![Figure 4.4: Example of the master branch.](image)
6. Choose a directory to store the local copy of codes and then click “Next”

![Figure 4.5: Example of the choosing directory.](image)

7. The code will be cloned and downloaded from the repository. This may take some time. When this is done, choose “Import existing projects” and then click “Next.”

![Figure 4.6: Example of the import projects.](image)
8. Import all of the projects and then click "Finish"

![Image of Import Projects window]

**Figure 4.7:** Example of all the projects which has been import.

9. The work-space should now contain all the projects (note - the actual projects may differ from following figure 4.8 depending on how recent the code is).

![Image of Eclipse workspace with Archi plugin]

**Figure 4.8:** Example of the plugin of Archi in Eclipse.

As shown in Figure 4.8, all the imported codes are in red, which means it needs a target platform to tell Eclipse how to run or debug Archi and what plug-ins to include and exclude in the configuration, using following steps [46]:

49
1. In Eclipse’s Package Explorer find the target file “com.archimatetool.editor.product/archi.target”, and open the file in the Eclipse Target Editor.

2. In the editor, at the top-right, click “Set as Active Target Platform”. Please be aware that in some cases this may read “Reload Target Platform”

3. Now let the eclipse download all the required dependencies, this may take a while so be patient.

Now all the the environment is set up for running or debugging the Archi. For running the archi go to the folder “com.archimatetool.editor.product/archi.product”, and open the file in the Eclipse Product Configuration Editor. In the editor, click “Run”.

### 4.2 Adding New Concepts to ArchiMate

As explained in the section chapter 3, the concepts that are proposed for this project are router, switch, interface switch, interface router and Routing Table. To add these concepts to Archi, a document from Archi, which was used as a reference for extending the model with new elements and attributes and is amended in appendix A. For the addition of these new concepts, the following procedures were done,

1. **Create the new element in the model**

   The new elements that are going to be added in the this project are router, switch, routing table, interface router, interface switch. To add an new element to Archi, it is necessary to create the element in the core of the model first as shown in following steps:

   (a) In the core model of tree select "archimate.core" and it will open the list of the elements that it has now. For adding new element click in the model and new child class. This will create the new blank EClass at the bottom of the tree. To add click in the model and new child.

   (b) Now that the elements has been added to the core model it is need to add they respective proprieties. The new elements are specialisation of the existing ones, so they need to inherit from their main element which has been specialised. Click on the properties windows and select “ESuper Types” to add the properties of new element. After adding all the elements, save the archimate.core and from the main generator, select “generate model code” and it will generate the java interface and the implementation of java class for the element as illustrated in Figure ???. Figure 4.9 shows the new elements proposed for this project.
2. **Add the new element to the Relationships rules:** Now, it is necessary to define the relationship between the new elements and the others. ArchiMate defines a set of rules used to represent the relationships between the elements. The definition of each type of relationship between elements was described in chapter 3. The relations are defined by letters where each letter represents a relation in ArchiMate as shown in Table 4.1 below.

<table>
<thead>
<tr>
<th>Letter</th>
<th>Relation</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Access</td>
</tr>
<tr>
<td>i</td>
<td>Assignment</td>
</tr>
<tr>
<td>c</td>
<td>Composition</td>
</tr>
<tr>
<td>r</td>
<td>Realisation</td>
</tr>
<tr>
<td>t</td>
<td>Triggers</td>
</tr>
<tr>
<td>g</td>
<td>Aggregates</td>
</tr>
<tr>
<td>o</td>
<td>Association</td>
</tr>
<tr>
<td>f</td>
<td>FlowTo</td>
</tr>
<tr>
<td>s</td>
<td>Specialisation</td>
</tr>
<tr>
<td>u</td>
<td>UsedBy</td>
</tr>
</tbody>
</table>

The relationship between the elements proposed and the existing ones are explained in the chapter 3.

3. **Add the new element to the ArchiMateModelUtils:** Now that the new elements have been created and the relationship between element is set up, add the new elements to the list of
technology layer that will appear to the user interface. From line 2-20, it will get all the elements created for the technology layer.

Listing 4.1: Adding new element to the list of technology layer

```java
public static EClass[] getTechnologyClasses() {
    return new EClass[] {
        IArchimatePackage.eINSTANCE.getNode(),
        IArchimatePackage.eINSTANCE.getDevice(),
        IArchimatePackage.eINSTANCE.getSystemSoftware(),
        IArchimatePackage.eINSTANCE.getTechnologyCollaboration(),
        IArchimatePackage.eINSTANCE.getTechnologyInterface(),
        IArchimatePackage.eINSTANCE.getPath(),
        IArchimatePackage.eINSTANCE.getCommunicationNetwork(),
        IArchimatePackage.eINSTANCE.getTechnologyFunction(),
        IArchimatePackage.eINSTANCE.getTechnologyProcess(),
        IArchimatePackage.eINSTANCE.getTechnologyInteraction(),
        IArchimatePackage.eINSTANCE.getTechnologyEvent(),
        IArchimatePackage.eINSTANCE.getTechnologyService(),
        IArchimatePackage.eINSTANCE.getArtifact(),
        IArchimatePackage.eINSTANCE.getRouter(),
        IArchimatePackage.eINSTANCE.getSwitch(),
        IArchimatePackage.eINSTANCE.getInterfaceRouter(),
        IArchimatePackage.eINSTANCE.getRoutingTable(),
        IArchimatePackage.eINSTANCE.getInterfaceSwitch()
    };
}
```

4. Define the user interface for the element

- Adding an icon to new elements:

  To add the new elements to the Archi code, first design the icons for new elements according to how it is suppose to appear while accessing the tool later.

  Next, add the preferred images (it has to be 16x16 pixel) of each element by importing it to the images folder of its corresponding class. The images that were imported for this project are depicted in the Figure 4.10.
For declaring these images, the following lines of code were added from line 1-5 for the 5 elements:

Listing 4.2: Code for adding images

1. `String ICON_ROUTER = ARCHIMATE_IMGPATH + "router.png";
2. `String ICON_SWITCH = ARCHIMATE_IMGPATH + "switch.png";
3. `String ICON_INTERFACE_ROUTER = ARCHIMATE_IMGPATH + "InterfaceRouter.png";
4. `String ICON_INTERFACE_SWITCH = ARCHIMATE_IMGPATH + "InterfaceRouter.png";
5. `String ICON_INTERFACE_ROUTING_TABLE = ARCHIMATE_IMGPATH + "routingTable.png";

Adding a graphical Eclipse framework Figure:

For adding a graphical eclipse framework, firstly you have to create a java class in the "com.archimatetool.editor.diagram.figures.elements". For each model object its need to create a class figure to represent the object and a interface to access to its proprieties. Listing 4.3 is the coding for create the view of the new elements to the user when they click in certain object for creating a new modelling process.

From the Listing 4.3, line 1-22 in the Listing 4.3 is for designing the element "interface router" when you click in the view of framework Archi to model a new process. Line 2-8 it will design a big rectangle for the new element. Inside that rectangle it also a small rectangle which is in line 13-23. Line 13-23 will design a rectangle with the x coordinate of the rectangle, the y coordinate of the rectangle and the width of the rectangle and the height of the rectangle.

The same code was used for design the interface of switch.

Listing 4.3: Creating the GEF of the element interface router

```java
public class InterfaceRouterFigure extends AbstractTextControlContainerFigure{
```
public InterfaceRouterFigure()
{
    super(TEXT_FLOW_CONTROL);
    fRectangleDelegate = new RectangleFigureDelegate(this, // 22 - getTextControlMarginWidth());
    fEllipseDelegate = new EllipseFigureDelegate(this);
}

@Override
/**
 * Draw the icon
 */
protected void drawIcon(Graphics graphics)
{
    graphics.pushState();
    graphics.setLineWidth(1);
    graphics.setForegroundColor(isEnabled() ? ColorConstants.black : ColorConstants.gray);
    Point pt = getIconOrigin();
    Rectangle rect = new Rectangle(pt.x, pt.y, 10, 10);
    // circle /rectangle
    graphics.drawRectangle(rect);
    // line
    graphics.popState();
}

/**
 * @return The icon start position
 */
protected Point getIconOrigin()
{
    Rectangle bounds = getBounds();
    return new Point(bounds.x + bounds.width - 14, bounds.y + 8);
}

@Override
public IFigureDelegate getFigureDelegate()
{
    int type = getDiagramModelObject().getType();
    return type == 0 ? fRectangleDelegate : fEllipseDelegate;
}
the big rectangle for the routing table element and from line 17-28 it will design three small rectangles in the left side of the big rectangle.

Listing 4.4: Creating the GEF of the element interface routing table

```java
public RoutingTableFigure() {
    super(TEXT_FLOW_CONTROL);
    fRectangleDelegate = new RectangleFigureDelegate(this, 22 - getTextControlMarginWidth());
    fEllipseDelegate = new EllipseFigureDelegate(this);
}

@Override
protected void drawFigure(Graphics graphics) {
    super.drawFigure(graphics);
    int type = getDiagramModelObject().getType();
    if(type == 0) {
        drawIcon(graphics);
    }
}

/**
 * Draw the icon
 */
protected void drawIcon(Graphics graphics) {
    graphics.pushState();
    graphics.setLineWidth(1);
    graphics.setForegroundColor(isEnabled() ? ColorConstants.black : ColorConstants.gray);
    Point pt = getIconOrigin();
    Rectangle rect = new Rectangle(pt.x, pt.y, 10, 10);
    //circle /rectangle
    graphics.drawRectangle(rect);
    graphics.drawRectangle(pt.x + 5, pt.y, 10, 10);
    //Restore the start of the graphics object
    graphics.popState();
}

/**
 * @return The icon start position
 */
protected Point getIconOrigin() {
```

55
Rectangle bounds = getBounds();
return new Point(bounds.x + bounds.width - 14, bounds.y + 8);

@Override
public IFigureDelegate getFigureDelegate() {
    int type = getDiagramModelObject().getType();
    return type == 0 ? fRectangleDelegate : fEllipseDelegate;
}

For designing the router and the switch, the same reference of code from the element node was used in the architecture. The code will design a cube. The code in the Listing 4.5 it was done based in the exiting element node [46]. From the Listing 4.5 line 28-31 it will return a rectangle that will be used in the line 7-22 for designing the cube.

Listing 4.5: Creating the GEF of the element for router

/**
 * Draw the icon
 */
protected void drawIcon(Graphics graphics) {
    graphics.setLineWidth(1);
    graphics.setForegroundColor(isEnabled() ? ColorConstants.black : ColorConstants.gray);
    Point pt = getIconOrigin();
    Path path = new Path(null);
    path.addRectangle(pt.x, pt.y, 11, 11);
    path.moveTo(pt.x - 0.2f, pt.y);
    path.lineTo(pt.x + 3.2f, pt.y - 3);
    path.moveTo(pt.x + 11, pt.y);
    path.lineTo(pt.x + 14, pt.y - 3);
    path.moveTo(pt.x + 11.2f, pt.y + 11);
    path.lineTo(pt.x + 14.2f, pt.y + 8);
    path.moveTo(pt.x + 3, pt.y - 2.8f);
    path.lineTo(pt.x + 14.3f, pt.y - 2.8f);
    path.moveTo(pt.x + 14, pt.y - 3);
    path.lineTo(pt.x + 14, pt.y + 8.2f);
    graphics.drawPath(path);
    path.dispose();
    graphics.drawRectangle(pt.x + 5, pt.y, 10, 10);
protected Point getIconOrigin() {
    Rectangle bounds = getBounds();
    return new Point(bounds.x + bounds.width - 18, bounds.y + 9);
}

- **Add a User Interface Provider:** This will display the new elements on the windows. For adding the user interface, a class is created on the package "com.archimatetool.editor.ui.factory.elements".

Listing 4.6: Creating the user interface for interface router.

```java
public class InterfaceRouterUIProvider extends AbstractArchimateElementUIProvider{

    public EClass providerFor() {
        return IArchimatePackage.eINSTANCE.getInterfaceRouter();
    }

    @Override
    public EditPart createEditPart() {
        return new ArchimateElementEditPart(InterfaceRouterFigure.class);
    }

    @Override
    public String getDefaultName() {
        return Messages.InterfaceUIProvide_0;
    }

    @Override
    public Image getImage() {
        return getImageWithUserFillColor(IArchiImages.ICON_INTERFACE_ROUTER);
    }

    @Override
    public ImageDescriptor getImageDescriptor() {
        return getImageDescriptorWithUserFillColor(IArchiImages.ICON_INTERFACE_ROUTER);
    }
}
```
Listing 4.7: Creating the user interface for interface switch.

```java
public class InterfaceSwitchUIProvider extends AbstractArchimateElementUIProvider {

    public EClass providerFor() {
        return IArchimatePackage.eINSTANCE.getInterfaceSwitch();
    }

    @Override
    public EditPart createEditPart() {
        return new ArchimateElementEditPart(InterfaceFigure.class);
    }

    @Override
    public String getDefaultName() {
        return Messages.InterfaceSwitchProvider;
    }

    @Override
    public Image getImage() {
        return getImageWithUserFillColor(IArchiImages.ICON_INTERFACE_SWITCH);
    }

    @Override
    public ImageDescriptor getImageDescriptor() {
        return getImageWithUserFillColor(IArchiImages.ICON_INTERFACE_SWITCH);
    }
}
```
return getImageDescriptorWithUserFillColor(IArchiImages.ICON_INTERFACE_SWITCH);

@Override
public Color getDefaultColor() {
    return ColorFactory.get(201, 231, 183);
}

@Override
public boolean hasAlternateFigure() {
    return true;
}

Listing 4.8: Creating the user interface for router.

public class RouterUIProvide extends AbstractArchimateElementUIProvider{
    public EClass providerFor() {
        return IArchimatePackage.eINSTANCE.getRouter();
    }

    @Override
    public EditPart createEditPart() {
        return new ArchimateElementEditPart(RouterFigure.class);
    }

    @Override
    public String getDefaultName() {
        return Messages.RouterUIProvider_0;
    }

    @Override
    public Image getImage() {
        return getImageWithUserFillColor(IArchiImages.ICON_ROUTER);
    }
}

@Override
public ImageDescriptor getImageDescriptor() {
    return getImageDescriptorWithUserFillColor(IArchiImages.ICON_ROUTER);
}

@Override
public Color getDefaultColor() {
    return ColorFactory.get(201, 231, 183);
}

@Override
public boolean hasAlternateFigure() {
    return true;
}

Listing 4.9: Creating the user interface for switch.

public class SwitchUIProvider extends AbstractArchimateElementUIProvider {
    public EClass providerFor() {
        return IArchimatePackage.eINSTANCE.getSwitch();
    }

    @Override
    public EditPart createEditPart() {
        return new ArchimateElementEditPart(SwitchFigure.class);
    }

    @Override
    public String getDefaultName() {
        return Messages.SwitchUIProvider_0;
    }

    @Override
    public Image getImage() {
        return getImageWithUserFillColor(IArchiImages.ICON_SWITCH);
    }
}
public ImageDescriptor getImageDescriptor() {
    return getImageDescriptorWithUserFillColor(IArchiImages.ICON_SWITCH);
}

@Override
public Color getDefaultColor() {
    return ColorFactory.get(201, 231, 183);
}

@Override
public boolean hasAlternateFigure() {
    return true;
}

The XML file of the elements that will be add to the core model of Archi is in the Listing 4.10. From the Listing 4.10, line 2-5 is the code to represent the element "router" in the windows view of Archi, line 6-9 is the code to represent the element "switch" in the windows view of Archi, line 10-13 is to represent the element "interface router" in windows view of Archi, line 14-17 is the code for represent the element "interface switch" in the Archi application and line 18-21 is the code to represent the element "routing table" in the application of Archi.

Listing 4.10: The XML file of the new elements add to the Archi
Listing 4.11 is the XML file for creating the new elements attributes when you click to add elements in the windows view for a new modelling process. The properties are “IP Address”, "Mac Address", "System Table", "Protocol" and "Routing Table".

Listing 4.11: The XML file of the new elements add to the Archi

```xml
<propertyTab
  afterTab="properties.tab"
  category="main.category"
  id="IP_address.tab"
  indented="true"
  label="%propertyTab.label.4">
</propertyTab>

<propertyTab
  afterTab="properties.tab"
  category="main.category"
  id="MAC_Address.tab"
  indented="true"
  label="%propertyTab.label.5">
</propertyTab>

<propertyTab
  afterTab="properties.tab"
  category="main.category"
  id="System_Table.tab"
  indented="true"
  label="%propertyTab.label.7">
</propertyTab>

<propertyTab
  afterTab="properties.tab"
  category="main.category"
  id="Routing_table.tab"
  indented="true"
  label="%propertyTab.label.8">
</propertyTab>
```
After doing all the above steps, running Archi from Eclipse you will see the result of the new element showing in Archi palette. Figure 4.11 shows the new elements in the Archi palette. The new elements added are the ones in light blue.

![Image of the new element into the palette of Archi](image)

**Figure 4.11:** Image of the new element into the palette of Archi

The final design of each element is represented in the following images. Figure 4.12 represents the router added to the ArchiMate Model with the proposed attributes which are name and IP address.
Figure 4.12: Final design of element router

Figure 4.13 represent the switch added to the ArchiMate model with the proposed attributes, which are name and MAC address.

Figure 4.14 represents the switch interface added to the ArchiMate model with the attribute name, a system table with all the interfaces actives and their respective IP and MAC addresses. It also includes a line to describe which protocol has been used for exchanging the packet.
Figure 4.14: Final design of element interface switch

Figure 4.15 represents the router interface added to the ArchiMate model with the new attributes, name, system table with all interfaces of router and their respective IP address. It also include a column for adding which protocol has been used to exchange packets.

Figure 4.15: Final design of element interface router

Figure 4.16 represents the routing table which can be used by the different routers connecting to the IXP for sending packets to different organisations. The routing table has a name and routing table with information about the network destination, netmask, gateway and a interface.
Figure 4.16: Final design of element routing table
5 Evaluation

Contents

5.1 Internet Exchange Point Model in Archi .............................. 69
5.2 Results Analysis ......................................................... 74
This chapter presents the evaluation for the proposed model. For the evaluation an IXP was modelled in Archi with the new elements added. Also two cases of processes running over the IXP were modelled for checking the alignment between the different domains in the proposed model. The last section is the analysis of the results.

5.1 Internet Exchange Point Model in Archi

Figure 5.1 is an example of IXP architecture which was explained in the chapter 2. Distinct organisations from different parts of the world can connect to it and share information or expose it over the exchange point. The organisation connected to the exchange point can share information by point-to-point or point-to-multi-point. A point to point communication is when for example AS 10 wants to share a specific information with AS 50. A multi-point communication when an AS 10 wants to share information with AS 30, AS 40 and AS 50. The different organisation can also add information to the IXP and check the information over the exchange point.

![Architecture of IXP](image)

Figure 5.1: Architecture of IXP

Figure 5.2 represents the Portuguese IXP implemented in Archi with new elements proposed on this thesis added. The design shows four organisations such as University of Lisbon, Ministry of Finance, Ministry of Social Security and Ministry of Justice who wants to connect to the IXP. The switch is where all the information exchange will go through. The element router is representation of each organisation.
5.1.1 Case studies

In this section it’s present the cases studies used for the evaluation of the IXP modelled in the Archi. There are several tools in the market that can be used for simulate the exchange packet between the organisations. These tools can verify the packets exchange, analyse the traffic and in case of a link failure, select the new path to deliver the packet to the right destination. If two organisations are sharing information and something happens in the network or a link fails the processes operated over the connection can be affected.

5.1.1.A Student Scholarship

This case shows what could happen to a process of a giving scholarship in a University if something goes wrong in the network. If student get access to study in a University but he does not have conditions to pay the school he can ask for a scholarship in his University. This process has to be repeated every year when the school starts according to the procedure shown in Figure 5.3.
If a student request for a scholarship the following steps would be carried out:

i. The student must be registered in the University for the current year that is applying for a scholarship.

ii. He/she has to have a document from Ministry of Finances that proves his/her economics status and deliver the documents to the University.

iii. After the University check the veracity of documents, it will grant the scholarship or declined it.

The University has to verify the documents that proves that the student has problems in paying the schools fees. To verify the veracity of the documents delivered, they have to request the finances for the information relate to the student to check the income in his family. The process of university connecting to the finances to check the required information is represented in the Figure 5.3.

Figure 5.4 is the process modelling for asking scholarship in the University. As you can see it is possible to model all the process from the infrastructure who supports the application until the business process who support this case.

As an example, if the University ask the information from the Finances Ministry to check the economics status of a student and if there is a latency on the network or if the network fails, this can affect the business process of the University. By doing this model it is possible to foresee if there is any change in the network which processes will be affected and how much will it cost for the change.
This case shows what are the process to access a criminal record if someone from social security wants to check the information for a specific person in order to know if this person is able to adopt a child or not. This information will decide whether the person will be accepted to adopt a child or not by the Ministry of Social Security. The process for checking the information will have the following steps,

i. The Ministry of Social Security will request Ministry of Justice to have access to the information.

ii. The Ministry of Justice will check if the person asking for that information has permission to access it.

iii. Then it will check the information inside a database with the current data from the user.

iv. Next it will generate a certificate about the person that the Ministry of Social Security is asking.

v. If the certificate contains any criminal record about the person the process of adoption will be rejected otherwise it will be accepted.

The diagram of the process for checking the criminal record is represented in the figure 5.5.
Figure 5.5: Description of the case 2

Figure 5.6: The model of case 2 in Archi
5.2 Results Analysis

After the evaluation it was identified that now it is possible to align the business process with the technology infrastructure. From the model proposed it is possible to see if an element of technology infrastructure is changed or removed, how it impacts the processes in business domain. For instance, if a link fails during the criminal record verification of someone or if the information has some latency it could leads to the wrong conclusion. As an example, if the router fails during the process of checking the record, the application of social security will no longer work and the process for the adoption will no longer be available to the applicants. The reds ones in the Figure 5.7 are the service that is going to be affected if the routes in the routing table are not working.

![Figure 5.7: Example of the impact in the process of adoption when the route fails or changed](image)

In the process for the attribution of scholarship if one of the routes in the routing table of Ministry of Finances fails it could delay the process of the attribution of the scholarship. If the routes fails, is possible to compute which applications can no longer run, what services can no longer be offered to the students. The process that can be affected are, university application, the student information, validation, scholarship accept, student reception.

Figure 5.8 is an example of the process that could not be available if the routes in the routing table changes during the connection between the University and Ministry of Finances. The red ones are the list of elements that will be affect by this change.
Figure 5.8: Example of the impact in the attribution of scholarship when a route fails or changed.

With the model before the addition of new concepts it was not possible to show what process can be affected if a route changes in the routing table and the elements of network provider who is supporting the different elements. With this model the organisation can see how critical a failure in the hardware can be and how robust the infrastructure should be to adapt these changes.
6.1 Conclusion

IXP is point of connection where a set of organisations can connect and share information according to the rules established within them. The concept of IXP has been explored from several years, to find new ways to improve the traffic exchange and protocols for the routing. This project uses the Design Science Research methodology to identify the problem, solution and how to implement it. The main goal of this project is to add more concepts to already existing modelling language ArchiMate. Today, there are several IXP points in the world, and some of the most successful IXP point is localised in Amsterdam, London and Frankfurt. In Portugal, the two major IXP are rSPtic implemented by AMA and ESPAP to connect all the public services and the GigaPIX which connects all the university and other telecommunication operators.

There are several modelling languages which can be used for modelling process, network and other types of activities like ArchiMate. After looking into today's state-of-art today, it was decided to implement an IXP with the modelling language ArchiMate. However, the ArchiMate framework does not have enough elements to represent the network. Hence, this project proposed new elements to add on the existing model such as router, switch, routing table, interface router and interface switch to overcome the shortcomings of existing elements. The router element was added as the element node alone cannot represent the IP addresses of all the different network elements connected in the IXP, likewise the switch element was added because the element node was not representing all the MAC addresses of the devices. The interface router element overcomes the technology interface to be able to represent the active interface of a router and the name of the protocol used for exchanging packets in the IXP. Routing table was also added as its existing counter part “element path” does not represent the list of paths and the IP addresses of packets that are to be send through the network.

The implementation of the proposed new elements was done in a tool denominated of Archi. Archi is an open source tool used for modelling and designing the different processes of ArchiMate. The Archi code was accessed through Git platform, which is used for software developments. After configuring eclipse IDE and getting access to the Archi code, the new elements were added to the model. With these new elements it was possible to model the IXP.

To evaluate the new elements added, firstly this work explained why IXP was modelled in EA language instead of using the network simulation tools. Next, a used case was considered for the evaluation, where the connection between University of Lisbon and Ministry of Finance was established first, and then verified if an entity from on organisation can access the resources from the other. For that the University will have to request and consult Ministry of Finance for the financial background check. To do so, University of Lisbon needs to establish connection to the IXP first, then access information from Finance Ministry to check student’s economic status to ensure that the University is granting scholarship to the deserving students.
Another used case was also considered for the evaluation, where the connection between the Ministry of Social Security and Ministry of Justice was established first, and then verified if the entity from Ministry of Social Security can access the resources of Ministry of Justice. For example, a applicant wants to deliver the documents at Ministry of Social Security for a process of adoption. For that the Ministry of Social Security have to consult the Ministry of Justice to verify the background of the applicant. To do so, Ministry of Social Security has to establish connection to the IXP and them connect to Ministry of Justice to access the applicants criminal records to ensure that the process of adoption is conceded to the applicants with no criminal issues.

With the case study the proposed meta-model was validated by showing how the IXP can be modelled in ArchiMate. The modelling of IXP in EA demonstrate how the changes in technology domain can have impact in process running in the business domain. In the used case, it was verified that a change in the routes or a fail in the hardware will have a big impact in the process. Also it was possible to compute which applications can no longer work, which services will not be offer to the clients and which process will have impact. Further uses of this relation could be of helping the company to make decisions, facilitating the introduction of new elements on the network and for clarifying how the new organisation will be connecting in the IXP.

To sum it up the proposed model allows the alignment between the strategic objectives of a organisation to the network provider who is supporting it and it allows to see the impact of changing the information on the network infrastructure.

The purpose of this work was to add new concepts in ArchiMate language model, and it was successfully achieved.

6.2 Limitations and Future Work

One of the limitations identified was in the implementing the new elements in the code of Archi because the information about how to add it was not very structured and the document used for adding it was from previous version of ArchiMate as the coding in the Git for the Archi has change significantly. Another limitation it was with the evaluation of the new elements introduced for the modelling language ArchiMate. The evaluation was done with the used cases but it was still not enough for evaluate the concepts introduced. With a real scenario of IXP where the elements introduced could be tested the results will be better.

In the future, more attributes or elements can be added to represent and cover wider concepts of network. Also defining the extension for the EA modelling language with new attributes and elements for other domains like applications can be consider.
Bibliography


Extending The Model with New Elements

A.1 Introduction

This tutorial provides an example of adding a new element concept to the ArchiMate model. This example element will be called "Business Element" and will belong to the "Business" ArchiMate layer. Ensure you have the latest code and open Eclipse with all the Archi projects available.

A.1.1 Create the element in the model

1. In Eclipse, open the archimate.ecore file in the "model" folder of the "uk.ac.bolton.archimate.model" plug-in:

2. In the tree of the Ecore editor, select the "model" node, right click, and select "New Child" and then "EClass". This will create a new blank EClass node at the bottom of the tree:
3. Select this new blank node and double-click it to open the Properties window.

4. In the Properties window add the name, “Business Element”.

5. As the element needs to inherit from the Business layer, it needs to inherit from the "BusinessLayer" interface. In the Properties window, select the “ESuper Types” row and click on the “...” button at the right of the row. A dialog box will appear. In the left column, select “BusinessLayerElement - ArchimateElement” and add it to the right column. Click OK:

6. The Properties window should look like the following:

7. Make sure you save the archimate.ecore file.

8. Open the archimate.genmodel file:

9. From the main “Generator” menu in Eclipse, select “Generate Model Code”. This will generate some files in the “src” folder - “IBusinessElement.java” (the Java Interface) and “BusinessElement.java” (the implementation Java class). Other files will be modified, too.

A.1.2 Add the element to the relationships rules file

As the ArchiMate language defines a strict set of rules for relationships between elements, its need to declare what relationships are allowed between our new element and the others. This is defined in the “relationships-2.0.xml” file found in the “model” folder. This is an XML file that is designed to be reasonably human readable at the expense of verbosity. An element is declared by its name in a “source” tag and allowable target elements are declared as “target” elements. Allowed relationships are set in the “relations” attribute. These are key letters like “o” and “c” and “f”. These are defined in the “relationships-keys.xml” file. For example, “a” represents the ”AccessRelationship”.

1. In Eclipse, open the “relationships-2.0.xml” file found in the “model” folder.

2. For this example, copy and paste the XML section for the “BusinessActor” section and rename it to the name of our new element, "BusinessElement". You will need to edit it to suit your rules. It should look like this:

```xml
<source element="BusinessElement">
  <target element="BusinessElement" relations="cfgostu" />
  <target element="BusinessActor" relations="cfgostu" />
  <target element="BusinessRole" relations="fiotu" />
  <target element="BusinessCollaboration" relations="fiotu" />
  ...
</source>
```
3. You will need to add "target" XML elements to the other "source" elements in the XML file so that relationship rules can be declared from these elements to the new element. The format will be:

```xml
<target element="BusinessElement" relations="cfgostu" />
```

A.1.3 Add the element to the ArchiMateModelUtils.java file

Now that the model element has been created, and the relationship rules defined we need to add it to the list of elements that will appear in the UI. The first task we need to do is add it to the "ArchiMateModelUtils.java" file.

1. In Eclipse, open the "ArchiMateModelUtils.java" file, found in the "uk.ac.bolton.archimate.model.util" package.

2. Edit the getBusinessClasses() method by inserting the following line amongst the others:

```java
IArchimatePackage.eINSTANCE.getBusinessElement()
```

A.1.4 Define the User Interface for the element

Now that we have added the element to the model plug-in, we need to turn our attention to the "uk.ac.bolton.archimate.editor" plug-in and provide some code to create the UI for the new element.

A.1.4.A Add an icon

1. Provide a 16x16 icon for the element. Save it as "business-element-16.png" and copy it to the "img/archimate" folder in the "uk.ac.bolton.archimate.editor" plug-in.

2. Declare the reference to the icon in the "IArchimateImages.java" file in the "uk.ac.bolton.archimate.editor.ui" package:

```java
String ICON_BUSINESS_ELEMENT_16 = ARCHIMATE_IMGPATH + "business-el
A.1.4.B Add a GEF Figure

1. Create a GEF (Graphical Eclipse Framework) Figure class for the element in the “uk.ac.bolton.archimate.editor.diagram.figures.business” package. Copy the “BusinessActorFigure.java” class and rename it to “BusinessElementFigure.java”.

2. Edit the constructor method in this class so that your new icon file is used:

   ```java
   figureDelegate.setImage(
   IArchimateImages.ImageFactory.getImage(IArchimateImages.ICON_BUSINESS_ELEMENT,16);
   ```

A.1.4.C Add a GEF Edit Part

1. Create a GEF (Graphical Eclipse Framework) Edit Part class for the element in the “uk.ac.bolton.archimate.editor.diagram.editparts.business” package. Copy the “BusinessActorEditPart.java” class and rename it to “BusinessElementEditPart.java”.

2. Edit the getFigure() method in this class so that your new figure class is returned:

   ```java
   return new BusinessElementFigure(getModel());
   ```

A.1.4.D Add a UI Provider

1. Create a UI Provider class for the element in the “uk.ac.bolton.archimate.editor.ui.factory.business” package. Copy the “BusinessActorUIProvider.java” class and rename it to “BusinessElementUIProvider.java”.

2. Edit this class to incorporate your classes from before:

   ```java
   public class BusinessElementUIProvider extends AbstractBusinessUIProvider {
   public EClass providerFor() {
   return IArchimatePackage.eINSTANCE.getBusinessElement();
   }
   }
   ```

   ```java
   @Override
   public EditPart createEditPart() {
   return new BusinessElementEditPart();
   }
   ```
```java
public String getDefaultName() {
    return "Business Element";
}

@Override
public String getDefaultShortName() {
    return "Element";
}

@Override
public Image getImage() {
    return IArchimateImages.ImageFactory.getImage(IArchimateImages.ICON_BUSINESS_ELEMENT_16);
}

@Override
public ImageDescriptor getImageDescriptor() {
    return IArchimateImages.ImageFactory.getImageDescriptor(IArchimateImages.ICON_BUSINESS_ELEMENT_16);
}
```

3. Finally, register this UI Provider. Edit the “ElementUIFactory.java” class by adding the following line to the ElementUIFactory constructor:

```java
registerProvider(new BusinessElementUIProvider());
```

A.1.5 Run Archi from Eclipse

Running Archi from Eclipse (see link) will result in your new element appearing in the context menus on the model tree and in the editor palette. The new element will appear in a diagram, and any relationship rules will be evident:
Create a ArchiMate Project in Archi

The tool use for the implementation of this project is Archi. Archi is a free and open source framework which is used for creating ArchiMate models and modelling sketches. The first version of framework Archi was introduced in the year of 2010, from this year until 2012 it was a part of a national project which the main propose was to support the EA in the United Kingdom higher education sector [47]. Since the year 2013, there have been many updates which lead to version 4.0 for supporting the ArchiMate 3.0. One of the main advantages of this tool is the possibility to store codes on GitHub (platform for the management of source code) when you need to implement new solutions. Following steps are used for creating a new ArchiMate model using Archi,

1. Select to create a "Create a new ArchiMate model" from the main "File New" menu
2. A model entitled ("new model") will be created with a default view where it is possible to add the elements according to the final solution.